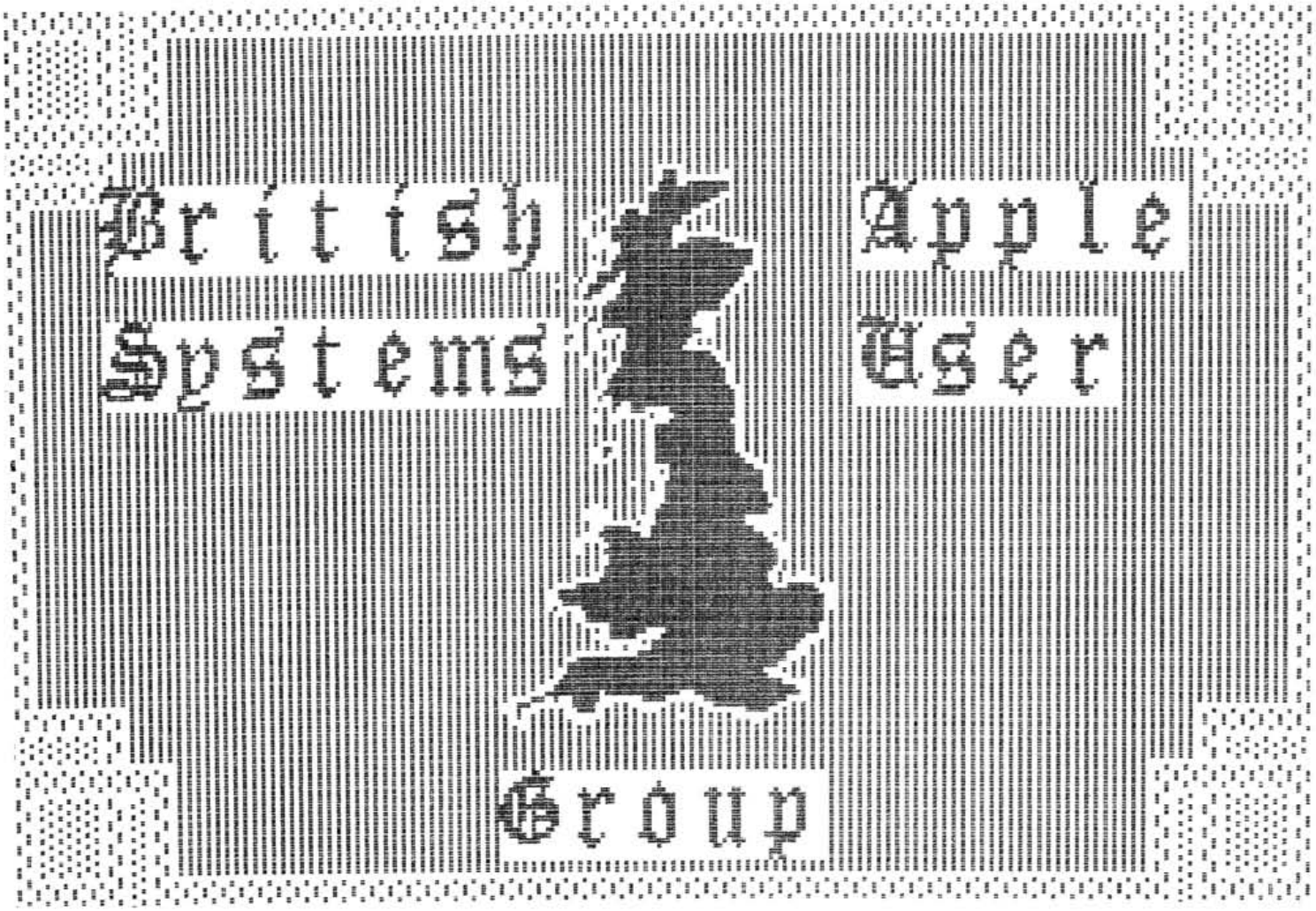
hard core

THE JOURNAL OF THE BRITISH APPLE SYSTEMS USER GROUP

AUGUST 1981 £1 VOLUME1 Nº4



DOS APPEND (NEW) FIX : ITT GRAPHICS FIXES : MORE SOFTWARE : ETC

hardcore

THE JOURNAL OF
THE BRITISH APPLE SYSTEMS USER GROUP
P.O.BOX 174 WATFORD WD2 6NF

EDITED BY DAVID BOLTON

CONTENTS

4 EDITORIAL

5 BEGINNERS PAGES — John Sharp

7 INTRODUCTORY DISK- A Problem — John Sharp

8 AMATEUR RADIO — Donald Maclean

11 A CAUTIONARY TALE- DOS Append errors and how to fix them — Neil McFerran

15 CHAIRMAN'S CORNER

16 PASCAL PAGES

18 PARK STREET HAPPENINGS — Tony Williams

21 I.R.A.M. REVIEW — Nik Spicer

22 THE ARK AND THE COMPUTER

24 ITT PATCHES for Automated Similations Games — Ian Trackman

25 ITT HI-RES HARDWARE MODIFICATION — David Bolton

26 APPLESOFT / PALSOFT DIFFERENCES — Ian Trackman

26 A LETTER FROM ITT

26 APPLE / ITT COLOUR CONVERSION CHART — Ian Trackman

28 HEX/DEC COMPETITION RESULTS

31 WARGAMERS VIEW of Microcomputer Games - Concluded — Robin Hood

34 SIG FOR THE DISABLED — Glyn Vernon

36 MINUTES OF THE A.G.M.

38 PRINTER INTERFACING Part 3 — Chris Murphy

41 THE SOFTWARE LIBRARIES — John Rogers

44 PROFILE - Frances Teo — Tony Williams

45 PIPPIN'S PAGE — Vernon Quaintance

46 EXPEDITER- A Review of Online System's Basic Compiler — Mike Walker

Cover- Hi-Res screen dump using an Epson MX 80 FT/2, of the Introduction to BASUG's first disk of software distributed to other user groups.

THE GO-BETWEEN

(CENTRONICS 737 VERSION)

Apple Writer is one of the most popular low-cost word-processing systems for Apple computers. The Centronics 737 printer is one of the most versatile dot-matrix printers available for the quality required for text-processing. Go-Between enables you to use all of the features of the 737 printer from within Apple Writer, so producing in combination an extremely cost-effective word-processing system.

Go-Between, written in machine-code for speed, produces properly justified text using the Centronics proportional character set. It also recognises a comprehensive range of printer commands embedded in your text-file, held in easy-to-use two-character codes (no complex control character sequences), which put these facilities at your disposal :-

* Left, centre, fill and right justification
* Left, right, top and bottom margin settings
* Switching between standard, proportional and condensed character sets
* Elongated characters
* Underlining (of any characters anywhere in a line)
* Optional printing of page headings
* Optional page numbering (1 - 99), starting at any number
* Single-sheet or continuous printing
* Full and half-line feeds in forward and reverse directions
* Single- and multiple-dot forward and reverse print-head movement
* Single and multiple line spacing in half-line increments
* Tabbing

All features function with any Country Character set.

During printing, embedded printer commands are displayed on the screen at the same time as they are executed by the printer.

A comprehensive 40-page manual is provided as part of the package.

Versions of Go-Between for Epson, Anadex and Paper Tiger printers will be available shortly.

Hard Core is "type-set" using Go-Between. "The ease of preparation was quite astonishing !" - David Bolton (Editor of Hard Core)

PRICE £26.50

BLUE CHIP SOFTWARE
10, Huson Close
London NW3 3JW

EDITORIAL

This issue, late as usual, is being prepared in the middle of the holiday season with all the additional problems that this causes. In particular, contacting people to get up-to-date information on local groups has not been possible, and therefore we are holding over this feature until the next issue. In the meantime, dates for your diary are as follows:-

The Park Street meetings will be on 1st Sept (Hands-On Software) and 6th October (Music Synthesis).

The new SOUTH LONDON group will hold it's inaugural meeting at 7pm on Tuesday 10th September in the rear hall of the Methodist Church, Corner of Lambton Rd & Worple Rd, Raynes Park. The Hall is about 2mins walking distance from Raynes Park (BR Southern Region) station with trains every 10 minutes from London. Further meetings are planned for 3rd Oct, 15th Oct, & 24th Oct. The contact is Warren Avrey - phone him on (01) 947 0254 or write c/o BASUG.

BASUG will be at the PCW show from 10-12 September, and we look forward to seeing both new and existing members there.

The next issue of HARDCORE will be in October, so please let us have copy as soon as possible. The following issue will be in December, and we would like to make this a 'Christmas Special', perhaps also issuing a disk of seasonal software!

Good news for those members who have joined recently - HARDCORE 1 & 2, which have been out of print, are being reprinted as a combined issue, and will be available very shortly. They will be sent to you automatically as soon as possible.

Finally, for those readers who are not members of the British Apple Systems User Group, we welcome all users and potential users of Apple Systems (Apple and ITT2020). Membership for the current year costs £10, including an introductory disk of software, subscription to HARDCORE (including back issues), access to bulk-buying facilities and special offers, and an ever-expanding range of meetings in various parts of the country. Simply send your cheque for £10 to P.O.Box 174, Watford, WD2 6NF.

David Bolton

It is the policy of BASUG that any commercial interest in the Apple by committee members should be declared. We therefore advise you that David Bolton, Frank Kay, and John Sharp have established a software company known as 'Microsource'.

BEGINNERS' PAGES

By John Sharp

MORE PEEKING AND POKING

In the last HARDCORE we began to look at PEEKing and POKEing. Before continuing it is worth a few minutes to consider how to save yourself a lot of work when you have a list of POKES to enter into a program.
If they are in a sequence as is most likely, why not let the computer do the work and use a loop to do it!-

```
10 FOR N = 768 TO 776
20 READ P
30 POKE N,P
40 NEXT
50 DATA 1,32,221,221,206,0,3,248,96
```

It will not have made much less work in this case but it will do so when there are many more POKES to do.

Now so far we have been mainly POKING. How can we use PEEK in a program to let us do something constructive. One way might be to see if you have an APPLESOFT or PALSOFT machine, i.e. an APPLE or an ITT2020. One way is to look into a location you know to be different; Ian Trackman suggests location 62447. If you PEEK (62447) and get a number other than 0 then you are running on an Applesoft in ROM machine.
PEEKing is often useful to find out other things in programs. It could be used to find out if you are in FLASHING or INVERSE or NORMAL mode when you want to PRINT a new string. This uses the location we dealt with last time. For example:-

```
10 X = PEEK(50)
20 IF X = 255 THEN PRINT "YOU ARE IN
NORMAL MODE"
30 IF X = 127 THEN PRINT "YOU ARE FLASHING
AT ME!!"
40 IF X = 63 THEN PRINT "YOU ARE INVERSE"
```

This may not be common occurrence but it could be useful if you were printing in different modes and wished to know which one you used last. Another case might be to test which key someone had pressed last. If you PEEK (-16384) then the value you get will get will be the ASCII value of the key pressed. This is illustrated by the following:-

```
10 FOR N = 0 TO 1000
20 X= PEEK( -16384)
30 IF X > 127 THEN PRINT CHR$(X);: POKE
-16368,0
40 NEXT N
```

The POKE -16368,0 is to reset the switch to read the next character typed in . You could use it to see which was the LAST CHARACTER typed on the keyboard, or indeed if any had been typed in whilst the program was running, without having to rely on the key being pressed at a specific time when the program requests it.

This suggests the following program for a simple reaction timer.

```
10 TEXT : HOME
20 FOR N = 1 TO RND (3) * 4000 : NEXT
30 PRINT "READY"
40 FOR N = 1 TO RND (3) * 4000 : NEXT
50 PRINT "STEADY"
60 FOR N = 1 TO RND (3) * 4000 : NEXT
70 PRINT "GO....."
80 POKE -16368,0
90 FOR N = 1 TO 300 : X= PEEK ( -16384) :IF X >
127 THEN GOTO 110
100 NEXT
110 PRINT "YOUR SCORE WAS ";1000 - N ; " CAN
YOU MAKE IT LARGER ?"
120 END
```

A few comments would help those who do not understand. Lines 20, 40 and 60 put a random delay into the program between each statement. The resetting of the keyboard in line 80 stops you cheating by pressing a key before the "GO" comes up.

POKING THE HI-RES SWITCHES

Have you ever tried to play a game and found instead of the correct graphics page the one from the graphics game you played before comes up. This is particularly common with the Integer games on the Software Distribution Library Disks. It means the switches are set wrongly for jumping between the various pages. These are POKE switches. If you read the APPLESOFT MANUAL it is confusing to say the least. An easier summary as follows is not foolproof because the switches are interdependent, but it usually works.

```
POKE -16304,0 ....TEXT TO GRAPHICS
POKE -16303,0 ....GRAPHICS TO TEXT
POKE -16302,0 ....GRAPHICS & TEXT TO FULL
GRAPHICS
POKE -16301,0 ....FULL GRAPHICS TO
GRAPHICS & TEXT
POKE -16300,0 ....PAGE 2 TO PAGE 1
POKE -16299,0 ....PAGE 1 TO PAGE 2
POKE -16298,0 ....LO-RES SWITCH
POKE -16297,0 ....HI-RES SWITCH
```

The last two are the most relevant ones if you have the trouble with being in the wrong type of graphics. Just press CTRL-C and type in the relevant POKE followed by a RETURN and rerun the program.

THE ESCAPE KEYS AND OTHER EDITING FACILITIES

If you are a beginner you have probably glanced at the REFERENCE MANUAL and thought "Yes I'll look at that some other time, maybe in a few years time". Well there are some pages you might understand that are not in hieroglyphics. For example the pages on the ESC codes for moving the cursor on pages 34 and 35. Although most AUTOSTART ROM users are familiar with ESC IJKM they are not always aware of the ESC DBAC set. They give you similar movements, but with one difference. If you press ESC ABCD then the next key you press actually works. It is not a "get you out of using the ESC key sequence ". If you are a poor typist and are continually mistyping CATALOG titles, you probably use the ESC IJKM keys to run up the Catalog and then along the row with the -> key. How many times do you forget to press R for the RUN twice or use some other means of getting out of the ESC sequence only to find you have typed UN MYPROGRAM and had SYNTAX ERROR come back at you. Well if you get into the habit of using ESC D as the last ESC character to move the cursor onto the line you want, there should be no such trouble.

Whilst on such facilities, do you know how to slow down or temporarily stop listings. A slow listing can be achieved by typing SPEED = 180 or some other number less than the normal 255. When you then LIST the letters will come out slowly. The lower the number the slower; with SPEED =1 they take a very long time. Remember to reset to SPEED = 255 before you run the program.

To slow the listing down so that a few lines come up at a time before stopping, use CTRL S. Type LIST and press RETURN as normal. Then put one finger on the CTRL key and keep it there. Then press the S key. Each time you press it the listing will either move or stop. In order to get back into BASIC type CTRL C. THIS WILL NOT WORK ON AN APPLE WITHOUT THE AUTOSTART ROM (including the ITT 2020) or if you are listing an INTEGER program.

BEGINNER BYTES BACK!

Wantage
Oxfordshire

Dear John,

Herewith I send you the promised beginners programs one of which I wrote to acquire some speed in typing being a non-typist. The other one was to aid my son's tables.

In both cases after RUNning the program questions are being asked, one in the form of a character which has to be typed in correctly to obtain the next one. The other one in the form of a multiplication of which the right answer has to be entered to proceed.

After the sequence has been succesfully ended a total time is given which is not very accurate but gives an indication of the progress made.

Unfortunately I could not find a more accurate timing routine and hope that one of the other members have a better proposal.

Yours sincerely,
Tom Winkel.

```
]LOAD TYPE
]LIST

10  FOR P = 0 TO 24
20 A =  INT (100 *  RND (1)): IF A < 32
OR A > 90 GOTO 20
30  HOME : VTAB (12): HTAB (20): PRINT
CHR$ (A)
40 PC = PC + 1: IF  PEEK ( - 16384) < 12
7 GOTO 40
50  GET B$: IF B$ =  CHR$ (A) THEN  NEXT
 P: GOTO 70
60  GOTO 30
70  PRINT : PRINT : PRINT "25 CHARACTERS
 IN ",1 *  INT (PC / 6.75)" SEC"


]LOAD MULTIPLICATION
]LIST

10  FOR P = 0 TO 19
20 A =  INT (10 *  RND (1)):B =  INT (10
 *  RND (1)):C = A * B
30  HOME : VTAB (12): HTAB (18): PRINT A
;" * ";B" = ";
40 PC = PC + 1: IF  PEEK ( - 16384) < 12
7 GOTO 40
45  INPUT D
50  IF C = D THEN  NEXT P: GOTO 70
60  GOTO 30
70  PRINT : PRINT : PRINT "20 MULTIPLICA
TIONS IN ",1 *  INT (PC / 6.75)" SEC"
```

THE INTRODUCTORY DISC - A PROBLEM

By John Sharp

If you joined in MAY/JUNE you may have been the victim of a brainstorm I had, with the result that your INTRODUCTORY DISK OR TAPE is not all it should be. For various reasons there were additions made which corrupted an otherwise good set of programs. What is suprising is that so few people have complained. To find out if you have a bad version run the GRANDAPPLE CLOCK and set the chimes. If they do not work then you have one of the faulty disks or tapes.

HOW TO CORRECT THEM

You can either return them for a corrected version, or carry out the following:-

POKE 33,33 to make sure that any strings you have to copy do not get extended by a lot of blank spaces (see HARDCORE 1 Beginners Page).

LIST the lines given

Use the ESC keys IJ or DB to get to the beginning of the line then use the -> key to run the cursor over the line and copy it into memory. Keep your eye out for syntax errors like a colon(:) being missing.

The relevant program lines are:-

HAUNTED CAVE	lines 1200 and 3000
CHICKEN	line 2200
ZOMBIE ISLAND	lines 150 and 159
FIVE GUESSES	lines 20 and 500
HIRES DEMO	lines 200 and 1000 (syntax)
HANGMAN	line 295
BLACK BOX	lines 36 and 1030
APPLE INVADER	lines 300 and 640
NIGHTMARE	lines 27 and 2070
GRANDAPPLE CLOCK	lines 170,350 and 800
SPACE MAZE	line 3010
MACHINE TO POKES	lines 115,310 and 1150
SHAPE DESIGNER	lines 92 and 150
SHAPE/DES ASSEM	lines 171,630 and 3140
SHAPER	lines 120,250 and 470
SHAPE -PROG	line 40
AUTOAPPLESOFT	lines 105 and 208
MINI ASSEMBLER	lines 100 and 195
SEARCH/ CHANGE	line 180

If I have caused you inconvenience I am sorry. We still cannot guarantee that the disk is bug-free, but if you do know of any problems please let us know. (How else can we sort them out?) If you still do not understand the programs, again please ask. An article can then be written by way of explanation and any suggestions incorporated as well.

AMATEUR RADIO

by Donald MacLean

So far, my ITT 2020 has worked in the 'shack' in six ways:

LOGGING. During contests, to tell me quickly if I've already worked a station. I did once concoct a program to help maximise points - there were multipliers for distance etc. - but I'm afraid I don't take contests too seriously. Countries from which I've received QSL cards are logged so that it's the work of seconds to check whether any Prefix is on the 'Wanted' list. The Apple 'Phone List' type of format works well for someone with my awful (human) memory, but it would be interesting to see some elegant 'custom' programs.

MORSE. The random groups generator ('Datong') is ideal for learners. Watching a micro print fast incoming morse is fun, but neither the 2020 nor the Tono 7000 (a dedicated Japanese micro, which I borrowed) cope well with degraded signals - or with eccentric human sending. Keyboard transmission of morse seems kinky somehow. Maybe because I learned to use a morse key (finger and thumb) before a typewriter (two fingers). Maybe just because I'm old-fashioned.

RTTY. Radio Teletype. Communicating from a keyboard to a VDU/Printer, perhaps thousands of miles away. Amateur RTTY is now standardised at 45.45 baud (bits/second) - a nice gentlemanly/ladylike pace - and many transceivers have dedicated RTTY signal-paths both ways, usually with Mark/Space frequencies of 1445/1275 Hz. There's a RTTY Repeater (Automatic relay), GB3PT, just south of Cambridge on UHF channel RB12, and the HF frequencies to monitor are 3.6, 7.04, 14.09, 21.1, and 28.1 Mhz and thereabouts. Two metre FSK is around 144.6 and AFSK about 145.3; 70 cms FSK at 432,6 and AFSK at 433.3 Mhz. Note that this mode is Verboten on Top Band - and that you must give station ID's by telephony or CW. (Chips to do both available now!) Run some RY's for the other guy to tune to, and don't send a string of LF's - a clear screen to you can be a room full of bumf to him. DL1WX at Langenfeld, Germany is a micro-processor 'Mailbox' on 14.097 Mhz - an intelligent device, but it wouldn't tell me if it was an Apple or not. (Use standard 45.5/170 RTTY). The ARRL transmit a News Bulletin from Washington at 2200 GMT daily on 14.093 Mhz under the call W1AW.

SATELLITES. The first purposeful program I wrote (in my comparably short Apple-era) list the AOS/LOS Times/ Azimuth/ Elevation (of Aerials) for orbits of OSCAR 8 which are accessible from SE England, given any Equatorial Crossing. Communicating with say New York, duplex, on UHF is fun! It also helps to have three hands. Sometime I mean to interface the ITT directly with the Antennae rotators - perhaps also with receiver tuning (which is digital anyway) to take care of Doppler shift.

SSTV. Slow Scan TV. Sort of facsimile. Chris Galfo's software includes a nice character generator. Somehow I can't enthuse about this mode - like keyboard morse: not my scene. Best place to find some SSTV is 14.23 Mhz.

ASCII-EXCHANGE. Now that our licence permits data transmission, I think that this is where it's at - or should be! Micro to micro - Apple to Apple. I've had some great QSO's while exercising my W6 licence in California (and dialled into the 'Bulletin-Boards'). American ASCII-Hams mostly seem to use 110 baud, though higher rates aren't rare, and almost all stay with 170 Hz shift. W1AW repeats its RTTY Bulletins in ASCII (110/170 on 14.093) around 22.15 GMT.

APPENDIX. Secretaries of relevant clubs / associations / groups:
(Addresses in the Call Book)

British Amateur Radio Teleprinter Group - GW3IGG
AMSAT-UK (Amateur Satellites) - G3AAJ
British Amateur Television Club, 13 Church St., Gainsborough, Lincs.

SALES AND WANTS

Members can advertise private sales and wants here, free of charge.

STARTER SYSTEM. ITT2020 (£450), Anadex Printer (£200), and Disc Drive with controller card **£950**
Phone [illegible]

EPSON TX-80B printer with interface, 6 months old Phone [illegible]

ITT 2020 - 48K with Integer Card and Programmers Aid No 1 **£450** or offers
Phone [illegible]

Sup-R-Term 80 column card	**£150**
9" Monitor	**£40**
Sharp PC12/11 Cassette Interface	**£7**

Mike Walker [illegible]

A CAUTIONARY TALE

(with apologies to Richard Adams & J R R Tolkien)

by Neil McFerran

Once upon a time in a far off island little Neil was sitting in the sunshine twiddling his nybbles and shifting his bytes to and fro, happy as a sandboy, when along came this wolf, most dapper in grey, all smiles and nimble toes.

"Why don't you collect some data for me, little sandboy?", said he, all of a simper.

"Oh certainly sir, with all your nice shiny teeth sir, I'll get all your bytes just as you like sir, and pack them up and parcel them just as you like for your later digestion, sir" I replied (it pays to be sociable to wolves, you see).

So began the Saga of The Terrible Trouble with The Database: first there was the hardware to be built to catch the Treasured Data, then the program to put it out to disk for safe keeping away from prying eyes and nosey noses and then the program to read it back from disk and then The Terrible Trouble really began. But before this happened there was a long summer of sweet smiles and lots of "How nice the System is" and "How easily it keeps track of everything" and a replete wolf with his daily ration of bytes and a satisfied Neil with everything running so sweetly, thanks to his Apple and a little ingenuity. But that, as I said, was before the black clouds gathered from all quarters and the cold fog obliterated the landmarks so we lost the Treasured Data and the wolf became lean and haggard from his lost bytes, with the disks a jumbled graveyard of bits of this and that all mixed together, not at all neatly formatted and ready for digesting.

Well, we'd better begin at the beginning and continue through the 'bad patch' until we get out into the summer's sun again here and now. It all started in the dim and distant past of long ago (about a year), when we began to collect data directly from some laboratory equipment with the Apple (write to the Ed. if you'd like to know how to connect your Apple to all sorts of equipment, perhaps he'll ask me to try to tell you - but that's another story). This data is for a large-scale clinical survey we're doing in Belfast which will help hospitals to diagnose various pathological conditions in the future using tests like the ones we have developed here. The only action required by the laboratory operator was to type in a number to the Apple so we could subsequently identify individual patients' test results which had been written automatically to the disk by the program. Since we were doing the tests 'double blind' we had no knowledge of which patients had what conditions and had only their numbers with which to identify them, so it was of utmost importance to be able to keep track of everything precisely.

To do this I arranged to have a sequential text file on the Apple called DATA.ALL to which the results of each day's tests were APPENDed. Having kept my eyes and ears open like a good child I knew all about the 'DOS 3.2 APPEND ERROR FIX' that Apple had issued (twice!), and included the fix before 'CLOSEing' 'DATA.ALL' after each APPEND. All seemed rosy in the garden, the disk was beginning to fill with the Treasured experimental data, each day's Treasure seeming to go into 'DATA.ALL' for safe keeping and to be easily viewable there all sparkling and fresh in its entirety. So I handed the System over to the person actually carrying out the laboratory tests and directed my attention elsewhere to organising an analysis program to correlate the laboratory results with the clinical findings. The 'Terrible Trouble' really started when we had been collecting data for about six months and had accumulated a considerable number of records in DATA.ALL. In order to know the total number of records to be recovered from DATA.ALL whenever it is to be interrogated I arranged that the first record be a counter for the total number of days on which DATA.ALL had been APPENDed to (each day's entry adds 54 variable length records to DATA.ALL). Then, on reading the first record, we have an account of the amount of Treasure we have hoarded, and know the total number of INPUTs from the disk to completely recover all of DATA.ALL. It was this small piece of program structure that eventually led both to much gloom and doom and also to our final recovery from 'The Terrible Trouble'. As I said, all seemed well at first, but the dreadful day dawned when I came to read the DATA.ALL file back and obtained an 'OUT OF DATA' error message from DOS after only two thirds of the expected records (from the contents of the first record) had been read back: everything seemed more and more gloomy as I examined the code for writing to DATA.ALL and then reading it back again - no errors became apparent no matter how hard I searched . So I resorted to examining the file directly using a Disk Utility program: from the CATALOG I knew there were some 145 blocks of data to be searched so I was rather overawed at the prospect of having to search each byte of each block for the End-Of-File marker (EOF) causing the 'OUT OF DATA' message. But undaunted I used the Utility program first to look up DATA.ALL's Track and Sector List (TSL) used by DOS to tell it where the file actually resides on disk (DOS Manual p.128): I found that there were actually 156 blocks there. At the time this didn't send cold shivers up my spine as it should have, instead I took comfort from the bit about inconsistencies in the length of TEXT files as indicated in the CATALOG (DOS Manual p.131). So I set off into the maze of sectors making up DATA.ALL with only the TSL to guide

me, I did have a rough idea where to start: about the same distance into the TSL as we had got with reading the total number of day's results before encountering the 'OUT OF DATA' error. Eventually, sure enough lurking in the middle of a block I found a foul little EOF, followed by more spanking new data, but it was recent data, very recent data and it was'nt right at the end of this SEQUENTIAL file ?..... But to try and recover Treasured data - any data - was a must.

The gloom by now had thickened so that day was hardly distinguished from night and steam was rising in long sighs from the weary disk drives (to say nothing of myself!). I tried to ignore the horrible phrases hanging in the air: "How am I going to recover the Treasure?" and "You don't really expect me to recall those patients to hospital, do you?" - this from the wolf lean and haggard, without a trace of his prancing footwork . But summoning my last reserves of strength and a friendly wizard, Paul, I set off again to the maze of the textfile, at least we knew where we were going this time - to the evil little EOF ."So how did it get there?" we asked ourselves and wrote a few little programs to have a look..... (see the listings at the end)

The murk was beginning to clear: it seemed that APPEND (with its fix) had been working fine all the time the textfile had been less than 128 blocks long. After that, instead of putting the first byte of the fresh data ON TOP of the EOF and thus extending the file by the extent of the fresh data, APPEND had been putting the first fresh data byte in the first location AFTER the EOF (see last month's Update). So, when we came to read the file, back execution had always halted when we hit the dreaded EOF, although freshly APPENDed data was hiding on the disk after the EOF. Our immediate response was to patch over the EOF marker on the disk in block #132 of DATA.ALL with an innocuous character (we used 'space') and hey presto with a shake of his staff the wizard had twinkled a whole day's data back from limbo! "But which day's Treasured data?", I hear you ask. Exactly, that's the difficulty, it's the very latest day's data not the one immediately following the day with the EOF at the end. In other words we really had lost a considerable part of the Treasured data because we had been APPENDing SEVERAL TIMES after having acquired the erroneous EOF in block #132 of DATA.ALL, with each successive APPEND starting at the same place (after the EOF in byte #93 in block #132) and overwriting most or all of the previously APPENDed data in the blocks after #132. (The amount overwritten depends on the actual number of bytes in each of the 54 records making up one day's data to be APPENDed, and this varies from day to day).

Using the Disk Utility we uncovered a swarm of EOFs lurking in block #145, a last memorial to the lost blocks condemned to limbo by the cankered APPEND, on patching over these the spirit of some much earlier day's work reappeared and was coaxed back to life: in the distant past the operator had been working overtime and APPENDed the equivalent of two day's results simultaneously and the second of these had survived relatively unscathed (see fig. 1: 'the next day').

Fortunately we had arranged that the operator receive a printed record of each individual result as it was obtained and so we were able to reconstitute the lost blocks gradually by writing a program to re-accept the data from the keyboard instead of directly from the laboratory equipment - boring, boRING.... Well at least it was better than recalling the patients to hospital for more samples of their bodies so that we could re-do the tests again - the wolf was beginning to smile, his walk was even becoming quite jaunty .

But every day we were processing new patient's samples, what were we to do to ensure that this would not happen in the future? We had already set up a fresh disk with a new DATA.ALL that had grown to 40 blocks, so should we restrict its size to <128 blocks and then start yet another fresh disk? This seemed very defeatist and uneconomical of disk space (127 out of, say 350 blocks -36%, very poor). So the only thing to do was to...."Bravely go where no man...." and delve into the dark dank depths of DOS in search of a new Treasure:

"Fix #2 for DOS APPEND."

While not having run the snivelling beast to his lair in pursuit of this Treasure we have, at least found a solution that will keep him whimpering at arm's length (see last listing - "....+FIX"). All that we are doing in this fix is to keep DOS's housekeeping in order: DOS has external file buffers whose number you may alter with the 'MAXFILES:' command. There is also one internal 'information buffer' which is used for whichever file DOS is currently working on. With a 48K system this buffer is at $B5D1-$B5F3 , these 46 bytes carry a range of information such as 'number of bytes in current external buffer to EOF', or 'number of 128-block units in file on disk'. Sometimes DOS makes mistakes with some of the entries in here, and it is these mistakes that produce the APPEND errors. But in other places in the buffer the correct information is still to be found, so our fix simply forces DOS to use the correct values in place of the erroneous ones. Now as to where DOS calculates these values.... the rumbling caverns, the squeaking fissures, your guess is as good as mine... (As I said: "whimpering at arms length" not "run to his lair"). We'll leave that to Apple, after all they wrote the System so they must have an annotated source listing (please..)... (Hint: if you change the code at $B33E -48k system- from LDA...., STA...., ADC.... to LDA...., STA...., CLC (NB this

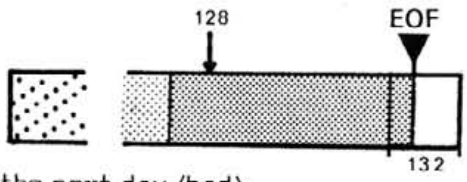
is the change), ADC.... this has the same effect as line 60040 in our fix below, but we haven't yet found a similar fix INSIDE the DOS for our line 60030).

So now all is well with the world, we're back in business, the wolf is snapping up all the bytes we can get him with the twinkle back in his eye and toes and the only thing remaining here is the homily:

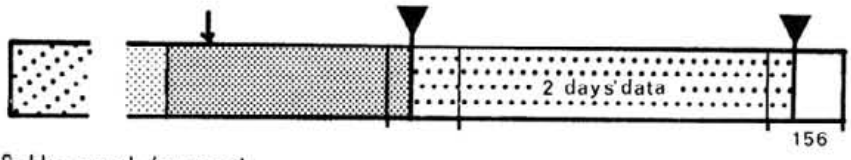
'Never rely entirely on the system, always do a little housekeeping of your own: if you expect to get 100 records back from the disk then first write that information out', this way you can keep count of just exactly how much Treasure you have, and
'Beware of APPEND', even in DOS 3.3 it's just the same - you get no warning of when it's about to fail, and it doesn't do it from the start.

Happy hunting!

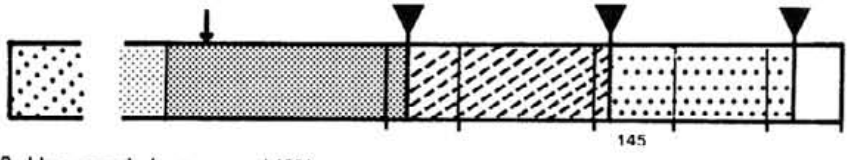
one fine day long ago
(when all was well with the world-just!),



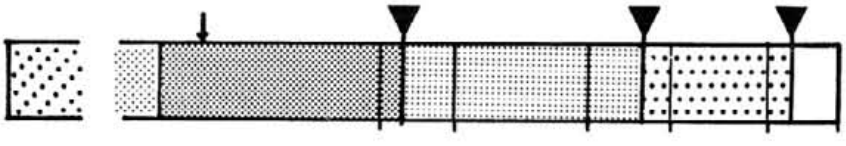
the next day (bad),



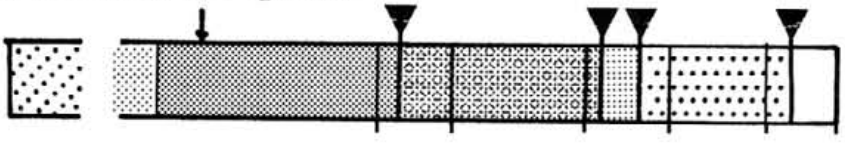
& the next (worse),



& the next (worse still),



& the next (worse again...)



Graphical representation of textfile DATA.ALL showing loss of data with successive APPENDs into blocks >128. (Numbers indicate block #).

PROGRAM LISTINGS

These 4 programs highlight the APPEND errors in DOS 3.2 and 3.3.

TESTFILE CREATOR initialises the sequential text file and sets the first record to have the value 'one' -ie: 1 data block in use IN TOTO.

The APPEND TESTERs read this first record and then APPEND as many whole blocks as you command, on completion the 1st. record is updated to contain the new value for the total number of blocks used.

FILE READER may now be used to check for an EOF in the body of the text file.

Running APPEND TESTER with TESTFILE >128 blocks long will create an APPEND error which cannot be removed with the fix, while running APPEND TESTER+FIX does not create an error

```
]LOADTESTFILE CREATOR
]LIST

0  REM
             PROGRAM
         TESTFILE CREATOR
1  REM

5  TEXT : HOME : VTAB 10: HTAB 12
     : PRINT "TESTFILE CREATOR": HTAB
     17: PRINT "PROGRAM"
10 D$ =  CHR$ (4)
20  PRINT D$;"OPEN TESTFILE": PRINT
     D$;"DELETE TESTFILE"
30  PRINT D$;"OPEN TESTFILE": PRINT
     D$;"WRITE TESTFILE"
40  PRINT "   1"
45  REM   WE HAVE NOW GOT A
        TEXT FILE ('TESTFILE')
        WITH ONE DATA BLOCK,
        CONTAINING THE VALUE '1'
46  REM
        THIS MAY NOW BE
        APPENDED TO IN ORDER
        TO PRODUCE THE ERRORS
50  PRINT D$;"CLOSE TESTFILE"
60  PRINT : PRINT "DO YOU WANT TO
      GO TO 'APPEND TESTER'    ST
     RAIGHT AWAY (Y/N)?";: GET A$
     : PRINT
70  IF A$ < > "Y" THEN  END
80  PRINT "WITH OR WITHOUT 'FIXES
     ' (W/N)? ";
90  GET A$: IF A$ < > "W" AND A$
      < > "N" THEN 90
100  PRINT : IF A$ = "W" THEN  PRINT
     D$;"RUN APPEND TESTER+FIX"
110  PRINT D$;"RUN APPEND TESTER"
```

```
0  REM
          PROGRAM
       APPEND TESTER
1  REM

10 D$ =  CHR$ (4)
20  FOR I = 1 TO 24:C$ = C$ + "HE
    LLO": NEXT I:C$ = C$ + "xxx"
    : REM
         NOW C$ IS 123 CHARS
         LONG - 24 'HELLO'S
         + 'xxx', LINE 110
         ADDS A 4 DIGIT
         SERIAL NO.
21  REM   WHEN PRINTED IN LINE
         120, A <CR> IS ADDED
         TO GIVE 128 CHARS.
         IN ALL.
30  PRINT D$ + "OPEN TESTFILE": PRINT
    D$;"READ TESTFILE"
40  INPUT NR
50  PRINT D$;"CLOSE TESTFILE"
60  TEXT : HOME : VTAB 10: HTAB 3
    : PRINT "YOU HAVE ";NR;" DAT
    A BLOCKS ON THE DISK."
70  PRINT : HTAB 2: INPUT "HOW MA
    NY MORE DO YOU WISH TO ADD "
    ;X
80  PRINT D$;"APPEND TESTFILE": PRINT
    D$;"WRITE TESTFILE"
86  REM    EXECUTE DOWN TO
         LINE 130 THE
         REQUIRED NO. OF
         TIMES.
90  FOR I = NR TO NR + X - 1
100 E$ =  STR$ (I): FOR J =  LEN
    (E$) + 1 TO 4:E$ = E$ + " ":
    NEXT J
101  REM  THIS MAKES A
         STRING(E$) OF
         LENGTH 4 WITH THE
         SERIAL NO. OF THIS
         PAIR OF RECORDS
         LEFT JUSTIFIED IN
         IT.
110 E$ = E$ + C$
120  PRINT E$: PRINT E$
130  NEXT I
140  PRINT D$;"CLOSE TESTFILE": REM
          ALL NEW RECORDS
          NOW APPENDED.
150  PRINT D$;"OPEN TESTFILE": PRINT
    D$;"WRITE TESTFILE"
151  REM  NOW UPDATE THE
         THE TOTAL NO. OF
         RECORD PAIRS INTO
         THE 1ST RECORD.
         SAME FORMAT AS IN
         LINE 100.
160 E$ =  STR$ (I): FOR J =  LEN
    (E$) + 1 TO 4:E$ = E$ + " ":
    NEXT J
170  PRINT E$
180  PRINT D$;"CLOSE TESTFILE"
190  PRINT : HTAB 14: PRINT "APPE
    ND COMPLETE": PRINT
200  PRINT "THE DATA NOW FINISHES
     IN BLOCK NO. ";I: PRINT
205  PRINT : PRINT : PRINT : PRINT
    "REMEMBER APPEND CRASHES IN
    BLOCKS >128          (128 I
    TSELF IS OK)": PRINT
210  PRINT : PRINT "DO YOU WANT T
    O GO TO 'FILE READER'      S
    TRAIGHT AWAY (Y/N) ?";: GET
    A$: PRINT
220  IF A$ = "Y" THEN  PRINT D$;"
    RUN FILE READER"
```

```
0  REM
        PROGRAM
     APPEND TESTER+FIX
1  REM
        SEE THE SUBROUTINE
        AT 60000 ONWARDS
        FOR A NICE FIX.
2  REM

10 D$ =  CHR$ (4)
20  FOR I = 1 TO 24:C$ = C$ + "HE
    LLO": NEXT I:C$ = C$ + "xxx"
    : REM
         NOW C$ IS 123 CHARS
         LONG - 24 'HELLO'S
         + 'xxx', LINE 110
         ADDS A 4 DIGIT
         SERIAL NO.
21  REM   WHEN PRINTED IN LINE
         120, A <CR> IS ADDED
         TO GIVE 128 CHARS.
         IN ALL.
30  PRINT D$ + "OPEN TESTFILE": PRINT
    D$;"READ TESTFILE"
40  INPUT NR
50  PRINT D$;"CLOSE TESTFILE"
60  TEXT : HOME : VTAB 10: HTAB 3
    : PRINT "YOU HAVE ";NR;" DAT
    A BLOCKS ON THE DISK."
70  PRINT : HTAB 2: INPUT "HOW MA
    NY MORE DO YOU WISH TO ADD "
    ;X
80  PRINT D$;"APPEND TESTFILE": PRINT
    D$;"WRITE TESTFILE"
85  GOSUB 60000: REM
         THIS IS THE FIX
         FOR APPEND'S
         POINTERS.
86  REM    EXECUTE DOWN TO
         LINE 130 THE
         REQUIRED NO. OF
         TIMES.
90  FOR I = NR TO NR + X - 1
100 E$ =  STR$ (I): FOR J =  LEN
    (E$) + 1 TO 4:E$ = E$ + " ":
    NEXT J
101  REM  THIS MAKES A
         STRING(E$) OF
         LENGTH 4 WITH THE
         SERIAL NO. OF THIS
         PAIR OF RECORDS
         LEFT JUSTIFIED IN
         IT.
110 E$ = E$ + C$
120  PRINT E$: PRINT E$
130  NEXT I
140  PRINT D$;"CLOSE TESTFILE": REM
          ALL NEW RECORDS
          NOW APPENDED.
150  PRINT D$;"OPEN TESTFILE": PRINT
    D$;"WRITE TESTFILE"
151  REM  NOW UPDATE THE
         THE TOTAL NO. OF
         RECORD PAIRS INTO
         THE 1ST RECORD.
         SAME FORMAT AS IN
         LINE 100.
160 E$ =  STR$ (I): FOR J =  LEN
    (E$) + 1 TO 4:E$ = E$ + " ":
    NEXT J
170  PRINT E$
180  PRINT D$;"CLOSE TESTFILE"
190  PRINT : HTAB 14: PRINT "APPE
    ND COMPLETE": PRINT
200  PRINT "THE DATA NOW FINISHES
     IN BLOCK NO. ";I: PRINT
205  PRINT : PRINT "REMEMBER THIS
     WILL ";: INVERSE : PRINT "N
    OT";: NORMAL : PRINT " FIX A
    PPENDS AT   >128 THAT HAVE B
    EEN SETUP WITH THE OTHERPROG
    RAM"
210  PRINT : PRINT "DO YOU WANT T
    O GO TO 'FILE READER'      S
    TRAIGHT AWAY (Y/N) ?";: GET
    A$: PRINT
220  IF A$ = "Y" THEN  PRINT D$;"
    RUN FILE READER"
230  END
59997  REM

        LOCS $42,$43 ARE A
     POINTER TO THE LAST
     USED FILE BUFFER (INFO)
     IMMEDIATELY AFTER FILE
     ACCESS.

59998  REM
      THE CONTENTS OF THE
      BYTE 30 LOCS INTO THIS
      BUFFER IS A COUNTER
      FOR UNITS OF 256-BLKS,
      IT MUST BE DUPLICATED
      IN BYTES $20 & $33
      (NOT SO >= 256 BLOCKS)
```

```
59999  REM
         BYTES $25 & $21 MUST
       BE IDENTICAL, BUT THIS
       IS NOT TRUE > 128 DATA
       BLOCKS IN FILE IN STD.
       DOS 3.2 OR 3.3
60000 AD =  PEEK (67) * 256 +  PEEK
     (66)
60030  POKE (AD + 20), PEEK (AD +
     30): POKE (AD + 33), PEEK (A
     D + 30)
60040  POKE (AD + 21), PEEK (AD +
     25): RETURN
65535  REM

            COPYRIGHT
          NEIL MC FERRAN
              &
          PAUL QUIGLEY
            Q.U.B.
```

```
0  REM
       PROGRAM
     FILE READER

3  ONERR  GOTO 200: REM
     SO WE CAN SHOW YOU
     WHERE TO LOOK ON THE
     DISK FOR THE 'EOF'.
5  TEXT : HOME : VTAB 10: HTAB 12
    : PRINT "TESTFILE READER"
10 D$ =  CHR$ (4)
20  PRINT D$;"OPEN TESTFILE": PRINT
     D$;"READ TESTFILE"
30  INPUT NR: REM
      THIS, THE FIRST RECORD
      SHOULD SERVE AS A
      COUNTER FOR THE
      TOTAL NUMBER OF
      PAIRED RECORDS IN
      'TESTFILE'
35  PRINT : HTAB 5: PRINT "BLOCK
     1 ALREADY OPEN, NOW"
40  FOR I = 2 TO NR
45  VTAB 14: HTAB 5: PRINT "READI
     NG BLOCK NO ";I;" OF ";NR:Y =
     X
50  INPUT A1$:B1$ =  LEFT$ (A1$,4
     ):B1 =  VAL (B1$)
60  INPUT A2$:B2$ =  LEFT$ (A2$,4
     ):B2 =  VAL (B2$)
61  REM    2*128-BYTE RECORDS=
           1 BLOCK ON THE DISK
70 X = B2
80  IF I = 1 THEN  NEXT I
90  IF B1 <  > B2 THEN 140
100  IF Y = X - 1 THEN  NEXT I: PRINT
     D$;"CLOSE TESTFILE": PRINT :
      HTAB 10: PRINT "TEST COMPLE
     TED SUCCESSFULLY": GOTO 500
101  REM    TO TRY APPENDING
            FRESH DATA
110  PRINT "SEQUENCE ERROR AT REC
     ORD PAIR #";I;: PRINT  CHR$
     (7);: FOR I1 = 1 TO 500: NEXT
     I1: PRINT  CHR$ (7)
120  NEXT I
130  PRINT D$;"CLOSE TESTFILE": END

140  PRINT  CHR$ (7);"REPLICATE R
     ECORDS DO NOT MATCH -";I;"-.
     "; CHR$ (7): NEXT I: END
200 X =  PEEK (222): IF X <  > 5 THEN
      PRINT : PRINT "ERROR CODE "
     ;X;", EXECUTION TERMINATED":
      END
201  REM    ERROR CODE 5 =
            'OUT OF DATA', ALL
            OTHER CODES STOP
            HERE.
210  PRINT "END OF DATA ENCOUNTER
     ED, AT DISK         LOCATION
      TRACK: "; PEEK (47084)
220  PRINT "          SECTOR: "; PEEK
     (47085): END
230  REM    LOCS 47084/5 ARE
            DOS'S TRACK AND
            SECTOR POINTERS IN
            A 48K SYSTEM.
500  PRINT : PRINT "DO YOU WANT T
     O GO TO 'APPEND TESTER'    S
     TRAIGHT AWAY (Y/N)?";: GET A
     $: PRINT
510  IF A$ <  > "Y" THEN  END
520  PRINT "WITH OR WITHOUT 'FIXE
     S' (W/N)? ";
530  GET A$: IF A$ <  > "W" AND A
     $ <  > "N" THEN 530
540  PRINT : IF A$ = "W" THEN  PRINT
     D$;"RUN APPEND TESTER+FIX"
550  PRINT D$;"RUN APPEND TESTER"
```

CHAIRMANS CORNER

This is my first Chairman's Corner, so I should like to begin by telling all of you what it has been like to see BASUG from the inside! It has been a somewhat sobering experience, although as I have served on BCS committees in the past, I suppose that the shock might have been anticipated! I have been enormously impressed by the obvious hard work and devotion given to the Group by your original Committee - this has involved countless hours of behind-the-scenes meetings in each other's homes, travelling to all sorts of places to introduce the Group to local clubs or to individuals who expressed interest, setting up the arrangements which allow members to save real money on items from blank cassettes to printers, organising the very successful members' meetings, and answering the flood of correspondence which arrives at P.O. Box 174 - this is currently running at upwards of 100 letters a week, mostly taking advantage of the club's facilities - more of this in a moment!

So, where is BASUG at this moment in time? We now have 600 members, from a standing start less than a year ago, which is an impressive record. The membership is growing at a rate of 20 a week currently, and our aim is to include every Apple/ITT user, whether personal, commercial or institutional, on our membership list before many more months have elapsed. You will find us in a prominent position at the PCW Show in September, and the Committee is spending a sizeable proportion of its time exploring ideas for spreading the word for BASUG. But, however energetically we do this, we are severely handicapped without the support and assistance of you - the average, involved, aware, intelligent, sophisticated and good-looking (I hope that works!) BASUG member! I mentioned before that the vast bulk of correspondence is involved in buying things through the Group, or ordering items from the Software Libraries. Well, this is terrific - but BASUG needs you, too! So, get powered-up! WRITE CONTRIBUTIONS FOR HARD CORE (even if only to encourage the others)! PERSUADE OTHER Apple/ITT OWNERS/USERS TO JOIN - YOU CAN TELL THEM FIRST-HAND what good value it is! FORM A LOCAL GROUP - if you write to BASUG, we can help with this - a local group doesn't have to have regular meetings, but it should help you all to increase your use and enjoyment of your computing. I am sure that a lot of you will have other ideas, so write to me at P.O. Box 174, and I will take the time to assist in developing these.

I will close expecting, as I am sure you do, great things of BASUG in the future. I thank the initial committee on your behalf for the great start they have given us, and hope that my next Chairman's Corner will allow me to report that the Apple seeds are sprouting everywhere! A final note - this will appear just before the PCW Show - come and see us, and help on the stand, even for a few minutes, if you can - we rely wholly upon your support.

Frank Kay.

pascal pages

```
PROGRAM LOGRAFS;
(* THIS PROGRAM PROVIDES LO-RES GRAPHICS
IN PASCAL. IT IS BASED ON BILL SHEPARD'S
ARTICLE IN APPLE ORCHARD, SPRING 1981.
   THE FULL UNIT WILL BE IN THE PASCAL
CONTRIBUTED SOFTWARE LIBRARY - WHEN IT'S
DEBUGGED! - BUT HERE'S SOME IDEAS TO GET
ON WITH.... *)
(* CONTRIBUTED BY WARREN AVERY, BASUG
                    S.W. LONDON GROUP*)

USES APPLESTUFF;
TYPE
  PLOTARRAY   =PACKED ARRAY [0..2047] OF 0..15;
   PLOTTYPE   =
     RECORD
       CASE BOOLEAN OF
         TRUE:  (POINTER :^PLOTARRAY);
         FALSE: (LOCATION:INTEGER);
      END (*CASE*);

  TEXTARRAY   =PACKED ARRAY [0..1023] OF CHAR;

  TEXTTYPE    =
    RECORD
      CASE BOOLEAN OF
        TRUE :  (POINTER :^TEXTARRAY);
        FALSE:  (LOCATION:INTEGER);
     END (*CASE*);

  PA = PACKED ARRAY [0..0] OF CHAR;
  MEMORYTYPE =
    RECORD
      CASE BOOLEAN OF
        TRUE : (POINTER  : ^PA);
        FALSE: (LOCATION :INTEGER)
     END (*CASE*);
VAR PLOTBUFFER   : PLOTTYPE;
  TEXTBUFFER   : TEXTTYPE;

  BASETEXT     : ARRAY [0..23] OF INTEGER;
  BASEGRAPHICS : ARRAY [0..47] OF INTEGER;
  MODETABLE    : PACKED ARRAY [0..255] OF CHAR;

  NULL,BLANK   : CHAR;
  I,Y,TOPY,
  CURSORX,
  CURSORY,COLOR :INTEGER;

PROCEDURE CLEAR (CH:CHAR);
BEGIN
FILLCHAR (TEXTBUFFER.POINTER^ [  0],120,CH);
FILLCHAR (TEXTBUFFER.POINTER^ [128],120,CH);
FILLCHAR (TEXTBUFFER.POINTER^ [256],120,CH);
FILLCHAR (TEXTBUFFER.POINTER^ [384],120,CH);
FILLCHAR (TEXTBUFFER.POINTER^ [512],120,CH);
FILLCHAR (TEXTBUFFER.POINTER^ [640],120,CH);
FILLCHAR (TEXTBUFFER.POINTER^ [768],120,CH);
FILLCHAR (TEXTBUFFER.POINTER^ [896],120,CH);
CURSORX:=0;
CURSORY:=0;
END (*CLEAR*);

PROCEDURE POKE(ADDRESS:INTEGER;CH:CHAR) ;
VAR BYTE   : MEMORYTYPE;
BEGIN
BYTE.LOCATION:=ADDRESS;
BYTE.POINTER^[0]:=CH;
END(*POKE*);


FUNCTION PEEK(ADDRESS: INTEGER):INTEGER;
VAR BYTE    : MEMORYTYPE;
BEGIN
  BYTE.LOCATION:=ADDRESS;
  PEEK:=ORD (BYTE.POINTER^[0]);
END(*PEEK*);

PROCEDURE ALLTEXT;
BEGIN
CLEAR(BLANK);
POKE (-16300,NULL);
POKE (-16302,NULL);
POKE (-16303,NULL);
TOPY:=0;
CURSORX:=0;
CURSORY:=0;
END(*ALLTEXT*);

PROCEDURE ALLGRAPHICS;
BEGIN
COLOR:=0;
POKE(-16298,NULL);
POKE(-16300,NULL);
POKE(-16302,NULL);
POKE(-16304,NULL);
CLEAR(NULL);
TOPY:=0;
CURSORX:=0;
CURSORY:=0;
END(*ALLGRAPHICS*);
```

```
PROCEDURE PLOT(X,Y:INTEGER);
BEGIN
X:=X MOD 40;
Y:=Y MOD 48;
PLOTBUFFER.POINTER^[BASEGRAPHICS [Y]+X+X+
                        (Y MOD 2)] := COLOR;
END(*PLOT*);

PROCEDURE HLIN (X1,X2,Y:INTEGER);
VAR BASE,X:INTEGER;
BEGIN
BASE:=BASEGRAPHICS [Y]+(Y MOD 2);
X:=X1+X1;
WHILE X <=(X2+X2)
  DO
    BEGIN
      PLOTBUFFER.POINTER^[BASE+X]:=COLOR;
      X:=X+2;
  END(*WHILE*);
END (*HLIN*);

PROCEDURE VLIN (Y1,Y2,X:INTEGER);
VAR BASE,Y:INTEGER;
BEGIN
BASE:=X+X;
FOR Y:=Y1 TO Y2
  DO
    PLOTBUFFER.POINTER^[BASEGRAPHICS[Y]+BASE+
                        (Y MOD 2)]:=COLOR;
END (*VLIN*);

PROCEDURE NORMAL;
BEGIN
   FOR I := 0 TO 31
     DO
       MODETABLE [I] := CHR (I);
   FOR I := 32 TO 95
     DO
       MODETABLE [I] := CHR (I+128);
 FOR I := 96 TO 127
     DO
       MODETABLE [I] := CHR (I - 96);
 END (*NORMAL*);

PROCEDURE LORESINIT; (*LOWRES INITIALIZATION*)
BEGIN
NULL := CHR (0);
BLANK:= CHR (ORD (' ') + 128);
PLOTBUFFER.LOCATION:=1024;
TEXTBUFFER.LOCATION:=1024;
TOPY   :=0;
CURSORX:=0;
CURSORY:=0;
FOR Y:= 0 TO 23 DO
  BEGIN
  BASETEXT [Y] :=(128 * (Y MOD 8)) + (40 * (Y DIV 8));
  BASEGRAPHICS [2*Y] := 2 * BASETEXT [Y];
  BASEGRAPHICS [(2*Y) + 1] := BASEGRAPHICS [2 *Y];
  END (*FOR*);
FOR I := 128 TO 255 DO
  MODETABLE [I] := CHR (I);
NORMAL;
END;(*INITIALIZE*)

PROCEDURE KALEIDOSCOPE;
VAR W,X,Y,Z  : INTEGER;
BEGIN
  FOR W:= 3 TO 8 DO
    BEGIN
    FOR X:= 1 TO 19 DO
      BEGIN
      FOR Y:= 0 TO 19 DO
        BEGIN
        Z:=X+Y;
        COLOR:=Y * 3 DIV (X + 3) + X * W DIV 12;
        PLOT(X,Z);
        PLOT(Z,X);
        PLOT(40-X,40-Z);
        PLOT(40-Z,40-X);
        PLOT(Z,40-X);
        PLOT(40-X,Z);
        PLOT(X,40-Z);
        PLOT(40-Z,X)
        END;
      END;
    END;
ALLTEXT;
END (*KALEIDOSCOPE*);

PROCEDURE TUNNEL;
VAR C : ARRAY [1..4] OF INTEGER;
    V,W,X,Y,Z : INTEGER;
BEGIN
FOR X:= 1 TO 4 DO
  C[X]:=(RANDOM DIV 2185) + 1;
WHILE  PEEK(-16384) < 127 DO
BEGIN
FOR X:= 3 DOWNTO 1 DO
  C[X + 1] := C[X];
C[1] := (RANDOM DIV 2185) + 1;
FOR X:= 1 TO 5 DO
  BEGIN
  FOR Y:= 1 TO 4 DO
    BEGIN
    COLOR:=C[Y];
    W:=Y * 5 +14 + X;
    Z:=39 - W;
    HLIN(Z,W,Z);
    VLIN(Z,W,W);
    HLIN(Z,W,W);
    VLIN(Z,W,Z);
    END;
  END;
END;
END(*TUNNEL*);



BEGIN (*DEMO LO-RES GRAPHICS*)
LORESINIT;
ALLGRAPHICS;
TUNNEL;
ALLGRAPHICS;
KALEIDOSCOPE;
ALLTEXT;
WRITELN('THAT''S ALL...');
END.
```

PARK STREET HAPPENINGS

By Tony Williams

Starting Machine Code

The Park Street meeting held on 12 May, 1981, was announced as "Starting" Machine Code, which was well understood by all the forty or so BASUG members who assembled, all, that is, except for the vociferous handful who departed in high dudgeon at half-time protesting that "We learned all this at primary school." Some primary school which teaches HEX but not reading (of announced talk titles). That aside, I think the majority of those who lasted through this at times hard-going exploratory talk found it worthwhile.

The talk was indeed tentative - as a precursor to a possible Introduction to Machine Code course which Ian would give sometime in the autumn if enough takers were forthcoming. After the collapse of BASUG's Pascal course, dropped because fifteen takers were not enough, Ian did well to soft-peddle this idea.

This talk then, began at the beginning and, as such, was more in the nature of an 'Introduction to Computing' combined with 'A Surgical Look at the Chips inside the Machine'. Graphically illustrated by a chance volunteer in the audience who just happened to be wearing a sweater knitted by Mum bearing an Apple emblem (Registered Trade Mark of Apple Computer Inc, I believe). Just the thing for the Trackman magic show, enabling him to illustrate the beginnings of it all in terms of warps and wefts. We then switched metaphors to take in counting on fingers, and then traffic light combinations. The naive spectator writing these notes got a lot out of it. In fact, everything the beginner wants to know about machine code, but is afraid to ask. What is all this about address busses, data busses, stacks, why Hex , why 8-bit, why 255 characters in a program line? That kind of thing. Most people at this meeting were afraid to ask too, despite Ian's jollying them along and laughing assurances that no one would giggle - out loud at least. When the silence was broken at last, the questions did tend to come from the pros who know all about machine code, er, sorry, assembly language, anyway, and were sitting there just to make sure that Ian got it right (very good-humouredly, I must add).

Mingling with the crowds afterwards this cub reporter heard that the talk was a splendid introduction, and that everyone would want to join a course and stump up the twenty or so pounds per head that Ian would want. What about it readers? Would you, could you attend a two or three day course somewhere around here sometime in the autumn? Give us your ideas.

We were glad to welcome at the club partners of the Milton Keynes Professional Workshop brought along by Peter Deakin. These are severely handicapped programmers using Apples to write commercial software as well as developing computer aids for the disabled.

Wordmongering

Members attending the July 7 meeting in Park Street were startled to hear the announcement from the newly introduced full committee that future meetings in the Old School are to be regarded as those of a local Hertfordshire group of BASUG and that volunteers from the audience were needed to step forward and take over. Stunned silence. The reasons for this are fairly obvious - with the national membership rising by forty a week - just imagine, say, that fine day when all six hundred decide to ... the mind reels.

Contrary to expectations the numbed members did not surge forward en masse to take up the reins of local committeehood, but, fear not, volunteers did present themselves in the course of the evening.

The evening was devoted to various presentations on word-processing packages available for the Apple. Ian Trackman ran us quickly over the Apple Writer's simple but robust qualities, and managed to slip in one or two words about his Go-Between disk - now at long last emerged from the satanic mills of Microsense. Frank Kay and Vernon Quaintance then set up a quick-talking double act taking us through the Super-Writer (£120), Easy-Writer which can do all kinds of sexy things on screen, including lower-case, aligning the text to match the output appearance, word wrapping and many more. This costs around £180 (excluding the card!). Vernon made us acquainted with - get this - Word Star on the Apple! This is apparently the star of micro WP packages although, as Frank did point out, it is about as simple to learn as flying a plane. Let's say that it has many, many features. This is all made possible by a lot of money. Some of it goes on the Z80 Softcard (Word Star runs under CP/M), more goes on the 80-column card and a thickish wad, about half an inch of fivers, goes on the Word Star software itself. Mind you, you get a lot for your money - including the ability to write your programs under Word Star - a useful thing to do because of the superb ease of editing. At work Vernon does all programming exclusively under Word Star in fact. No aeroplane pilot he, Vernon says a reasonable secretary can start doing useful things with Word Star within a week or so.

A planned afterthought came from Frank Udy on Apple Pie, an unpretentious but interesting low-cost WP package. This easy-to-use program also likes you to have an 80-column card.

The evening ended with the usual disgusting spectacle of people taking advantage of our low prices by buying disks, Apple Orchards, tee-shirts, etc.

I.R.A.M. REVIEW

By Nik Spicer

(Editor- Nik Spicer writes commercial software, almost all of which involves extensive file handling. We therefore asked him to review T.Tse's Indexed Random Access Method - see Hard Core #3 - from an expert's viewpoint)

In any situation where information is held in disk records, fast access is normally achieved by creating an index table containing a list of strings, each string identifies a particular random access record. To find a record you have to match an input string to a string in the table, and by inferrence from the positon of the string, or by looking at a number held as part of the string, the required record number is found. This works fine - in a simple application - so why use t. tse's routine?

Let's compare memory usage and timings:

String array 330 elements X 16 characters each = 6270 bytes
Tse system 330 elements X 16 characters + routines = 5346 bytes

No great saving there, but remember that string manipulation on the string array, using BASIC, will create dead bytes and possibly force a garbage collection.

Longest search time (unsuccessful search):
String array as above 1.8 seconds
Tse routine as above < .3 seconds (difficult to measure)

So certainly the search time is very good.

Get index from disk timings:
String array as above 22.4 seconds
Tse routine as above 7.27 seconds including loading routines

That is worthwhile, but remember that loading need happen only once a run.

Since both systems go on to load random access files (a process that takes at least 3 seconds) the speed gain in searching may not be too important, so the advantage appears to lie in memory conservation. Tse's routine will not use more memory than the defined index area.

However, using an index array normally gives you the advantage of sorting the records if the record number is held as part of the array. If a BASIC sort is used this process will create a mass of dead bytes at least equal to the original array, and is also painfully slow. Using a binary sort solves this problem since it is very quick and leaves memory size untouched. Tse's routine will not sort directley and I feel that this is an important ommission. Since Tse's index keys do not hold record numbers independant from the implied record of position in file, sorting them would be a problem.

Being able to sort the "keys" has a profound effect on search time using a string array with record number included, changing search time from 1.8 seconds to equal or faster than Tse's routine. the search process is well known, and my own routine is:

```
70 F% = X /2 :E% = X +1 :S% = 1: C% = 0
72 IF R$ = LEFT$(A$ (F%), LEN(R$)) THEN RETURN
73 C% = C% + 1: IF C% = 12 THEN C% = -1: RETURN
74 IF R$ > LEFT$(A$ (F%), LEN(R$)) THEN S% = F% :F% = F% + ((E% - S%) /2) :GOTO 62
76 E% = F% : F% = F%-((E% - S%) /2): GOTO 72
```

Where X = number of elements in array
R$= search string
A$() = array of key strings in format A$(I) = "AAAAAAAAAAArec#"
If C% returns = -1 then no match, otherwise A$(F%) = key string matched.

The worst not found case (using 1000 keys) loops through the routine only 10 times, and equals the machine code routine in speed - logic versus brawn.

The one requirement is a fast sort such as Ampersort.

The other disadvantage I found while using the Tse system, and this is arguably relevant only to a programmer, although I think not, is that there is no direct way of viewing/printing out the "keys". This would be simple with an array.

The immediate solution that occured to me, but there may be better ones, is the following adaption to the Demo program:

```
8 C$ = "": CX = PEEK (106) *256 + PEEK(105)+2
44 PRINT "6. DISPLAY KEY CODES"
50 VTAB 19: PRINT "WHICH (1 - 6)"
65 IF A$ < "1" OR A$ > "6" THEN 55
70 ON VAL(A$) GOSUB 1000,2000,3000,4000,5000,6000
6000 POKE CX,KEYLN: BA = INDEX + KEYLN
6010 POKE CX + 2, BA/ 256: POKE CX +1,BA - PEEK(CX+2)*256
6020 IF ASC(A$) = 0 THEN PRINT "FINISHED";:CALL - 676:
RETURN
6030 PRINT "KEY REC # " (BA - INDEX) /16 SPC (5 - LEN
(STR$ ((BA - INDEX)/16)))"= "C$
6040 BA = BA + KEYLN: GOTO 6010
```

This routine steps through the keys one at a time, pointing the variable C$ at each one so that C$ can then be printed.
This routine cannot be used for sorting except in a very crude way.

To sum up I believe that Tse's routines may point the way to a much more comprehensive system, but the existing routines have only limited advantage over a standard sorted array system. I would like to see a sort facility (which would involve adding 2 bytes to each "key" to denote record number (0 - 32767 integer)), and a machine code adaption to point a variable at the found "key". This access to the "key" removes any need to store the "key" in the main file.

Another feature worthwhile adding would be a "wildcard" character within the search string, enabling groups of records to be retrieved and displayed. The modification for this would be quite easy, and if a new entry routine was included which enabled a "continue search from last found key position + 1" the system would get very useful indeed.

-+-

(Editor)- We asked T.Tse (the author of I.R.A.M.) for his comments, and here they are, word for word!

I have read with some interest Nik Spicer's review of my article on IRAM for the Apple II. My first impression was that I should never have started that article !!! However, I would still like to make some eye-opening comments.

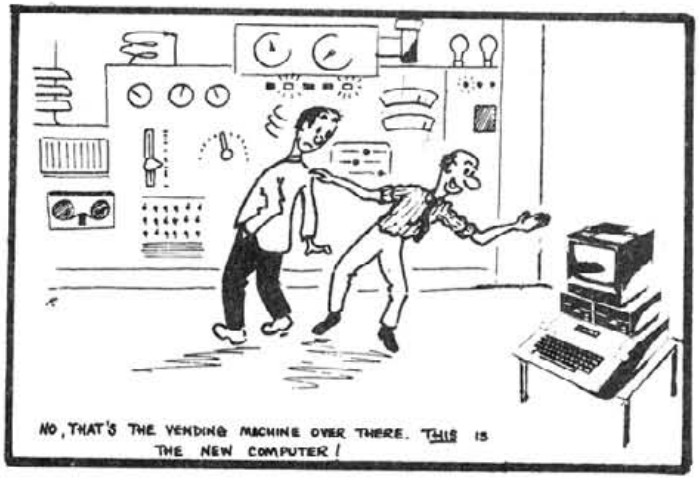
a) A very similar machine code routine found its way into a commercial application as far back as NOV 1979 - long before anything like Amper-Sort and Amper-Reader.
b) The IRAM article is the first complete and superbly written application program that Hardcore has published.
Just like Nibble and others, Hardcore can now think in terms of a regular feature article !!
c) Keyed Random Access Methods are now commercially available for the Apple II. One is Super-KRAM (United Software of America) which allows hierarchial structure to the database and various fast key retrieval and sorts. It also occupies approximately 16K of valuable RAM.

Finally a fix for those who have tried modifying KEYST and failed. For KEYST = LLHH you also need to perform the following:

9521=LL
9523=HH
9576=LL HH
9586=LL HH

I am very sorry that this last item was overlooked.

T.Tse.



THE ARK AND THE COMPUTER

And the Lord said unto Noah, "Where is the Ark which I have commanded thee to build?"

And Noah said unto the Lord: "Verily, I have had three carpenters off ill. The Gopher wood supplier hath let me down ... yea, even though the Gopher wood hath been on order nigh upon 12 months. What can I do, O Lord?"

And God said unto Noah: "I want that Ark finished even after seven days and seven nights."

And Noah said: "It will be so."

And it was not so. And the Lord said unto Noah: "What seemeth to be the trouble this time?"

And Noah said unto the Lord: "Mine subcontractor hath gone, alas, bankrupt. The pitch which thou commandest me to put on the outside and the inside of the Ark hath not arrived. The pipefitter hath gone on strike. Shem, my son, who helpeth me on the Ark side of the business, hath formed a rock group with his brothers Ham and Japheth. The canvas, although on hand, is not the right colour. Lord, I am undone."

And the Lord grew angry and said: "And what about the animals, the male and female of every sort that I ordered to come unto thee to keep their seed alive upon the face of the earth?"

And Noah said: "They have been delivered unto the wrong address but should arriveth on Friday."

And the Lord said: "How about the unicorn and the fowls of the air by sevens?"

And Noah wrung his hands and wept, saying;"Lord, unicorns are a discontinued line, thou canst not get them for love nor money. And fowls of the air are sold only in half-dozens... the peacocks even then are on back order for weeks to come. Lord, thou knowest how it is."

And the Lord said: "Rest my son. In my wisdom I should have known this would follow when I listened to Satan and installed a computer."

(We found this in Pete and Pam's recent Dealer Newsletter. Thanks, P&P, for allowing us to reprint this - it deserves a wide audience.)

HELP WANTED- RESEARCH PROJECT

University of Lancaster,
Department of Marketing,,
School of Management
and Organisational
Studies,
Gillow House,
LANCASTER LA1 4YX.

Dear Mr Bolton,

I am a postgraduate student, and am engaged in a research project on the problems encountered by business users of Microcomputers.

The purpose of this project is to publish a report which will guide small businesses contemplating a move into microcomputing, and as this will be based on the experience of actual users we believe it will be a very worthwhile publication.

It is therefore necessary to talk to current users and hence, could you please let me know the names and addresses of those members who you think would be willing to complete a postal questionnaire or take part in an interview lasting approximately twenty five minutes. All information will be treated with confidence, and each respondent will be sent a free copy of the report.

I do hope you are able to assist and look forward to hearing from you.

Yours sincerely,
Brian Kelly.

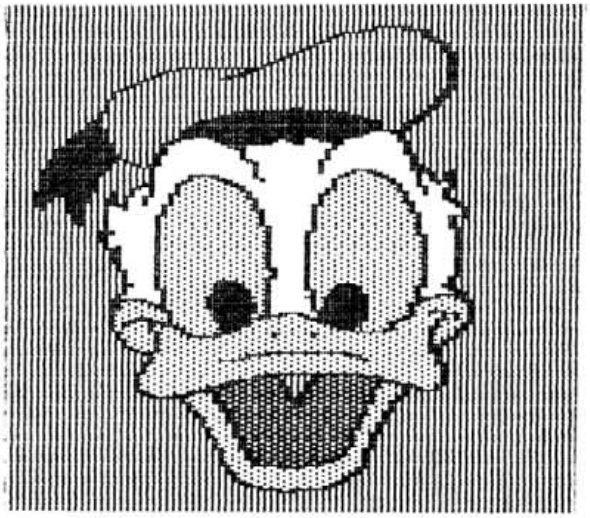
xxx We cannot release members names without their approval, and therefore ask that those of you willing to help write direct to Mr Kelly.

Apple Xperts

We are interested in Apple programmers who might be available to do part time, fixed price, development jobs. Please send relevant details to box number below.

We are also particularly interested in any applications software, developed by such people, where we would market, support and pay licence fees to authors.

BOX E1, BASUG, P.O.Box 174, Watford, WD2 6NF.



"DATESTONES OF RYN"
"TEMPLE OF APSHAI"
"MORLOC'S TOWER"
"RESCUE AT RIGEL"

ITT 2020 CONVERSION PATCHES

By Ian Trackman

As all of the programs are written in Applesoft, the Basic hi-res drawing commands will work on ITT 2020s, although the screen display uses only the left-most 279 vertical display columns instead of the full 359 columns available on the ITT. This results in a slightly condensed image and anyone troubled by it - and who has the patience - is welcome to go through the programs, converting all horizontal plotting references by a factor of 359/279 !

hd

More important is the fact that all of the programs use a short machine-code routine to clear that part of the high-resolution page containing the map. When used on an ITT, it wipes out the wrong area. I have written a replacement routine and for those interested in how it works, here is its source-code :-

```
        LDA   $E6       ; HI-RES PAGE (HRG OR HGR2)
        STA   $27       ; HBASH (SCREEN MEMORY MAP)
        LDA   #0        ; INITIALIZE Y
        STA   $26       ; & SET HBASL TO 0
        LDX   #$C0      ; FOR 192 HORIZONTAL LINES
LOOP1   LDA   #0        ; BLACK
        TAY             ; & RESET HORIZONTAL COUNTER
LOOP2   JSR   $F4D3     ; SWITCH ON 9TH BIT
        LDA   #0        ; BLACK AGAIN
        STA   ($26),Y   ; POKE IT ONTO SCREEN
        INY             ; NEXT PIXEL
        CPY   #$15      ; LINE OF 21 PIXELS YET ?
        BNE   LOOP2     ; NO, SO CONTINUE THIS LINE
        JSR   $F536     ; YES, SO DROP ONE (GRAPHICS) LINE
        DEX             ; & DECREMENT LINE COUNTER
        BNE   LOOP1     ; NEXT LINE IF NOT DONE
        RTS             ; ALL DONE & BACK TO BASIC
```

With a small amount of adaptation, this routine can be used as a general routine for fast filling of any size of high-resolution rectangle in any colour anywhere on the screen. Please note that if one edge of your rectangle falls on a "9th bit" line, (as our patch does), you will have to clear that line separately. It's easier to do this directly from Basic (as in the patched Line 4001 below) rather than by writing a special machine-code routine for the purpose.

hd

Please note that since ITT and Apple high-resolution graphics are different, my routine won't work with Apples. Instead, adapt the routine already in the programs.

The decimal DATA of the sub-routine which has to be POKE'd into the programs is as follows :-

DATA 165, 230, 133, 39, 169, 0, 133, 38, 162, 192, 169, 0, 168, 32, 211, 244, 169, 0, 145, 38, 200, 192, 21, 208, 244, 32, 54, 245, 202, 208, 235, 96

I'll give the line references for each program in a moment.

The other change to be made arises from the hi-res colour differences between ITTs and Apples. There's a conversion table in this issue.

DATESTONES OF RYN

Make the following changes in the sub-program "DATES" :-

LINES 35 - 37

(DATA 169, 32 ....)

Delete these two lines and substitute a new line 35 :-

35 DATA [data as listed above]

LINE 40

(FOR I = 912 TO 969 : READ J : POKE I, J : NEXT)

Change 969 to 943

LINES 104 and 145

Change HCOLOR = 5 to HCOLOR = 6

LINE 680

Change HCOLOR = 6 to HCOLOR = 5

LINE 4001

(CALL 912 : GOSUB 80)

Change this line to :-

4001 CALL 912 : HCOLOR = 0 : HPLOT 190, 0 to 190, 191 : GOSUB 80

TEMPLE OF APSHAI

Make the following changes in the sub-program "DM" :-

LINES 35 - 37, 104, 145 and 680

Make the same changes as for Datestones of Ryn.

LINE 40

(FOR I = 7936 TO 7993 : READ J : POKE I,J : NEXT)

Change 7993 to 7967

LINE 4001

(CALL 9736 : GOSUB 80)

Change this to :-

4001 CALL 9736 : HCOLOR = 0 : HPLOT 190, 0 TO 190, 191 : GOSUB 80

MORLOC'S TOWER

Make the following changes in the sub-program "MORLOC6" :-

LINES 23 - 25

Delete these three lines and substitute a new line :-

23 DATA [data as listed above]

LINE 26

Change 969 to 943

LINES 104 and 145

Change HCOLOR = 6 to HCOLOR = 5

LINES 680, 836, 898, 899

Change HCOLOR = 5 to HCOLOR = 6

RESCUE AT RIGEL

Make the following changes to the sub-program "RESCUE 6.18" :-

LINES 23 - 26, 104 and 145

Make the same changes as for Morloc's Tower.

LINE 893

Change HCOLOR = 6 to HCOLOR = 5

I haven't found it necessary to alter Line 4001 in Morloc or Rigel but it's possible that I didn't explore all of the rooms !

(Editor) - Have any members converted any other programs for ITT 2020s ? If so, let's hear about the patches.

ITT 2020 HI-RES MODIFICATIONS

By David Bolton

The last issue of Hardcore referred to a modification by Ron Davies to solve the Apple/ITT incompatibilities, which invloved cutting the tracks on the PCB to Bits 8 & 9, and installing a switch to optionally copy bits 6 & 7 into these.

A similar modification has now been advised by DDP, which does not involve permanent modifications, as follows:-

ITT 2020 modification to suppress "Tram lines" in Apple Hi-res programs.

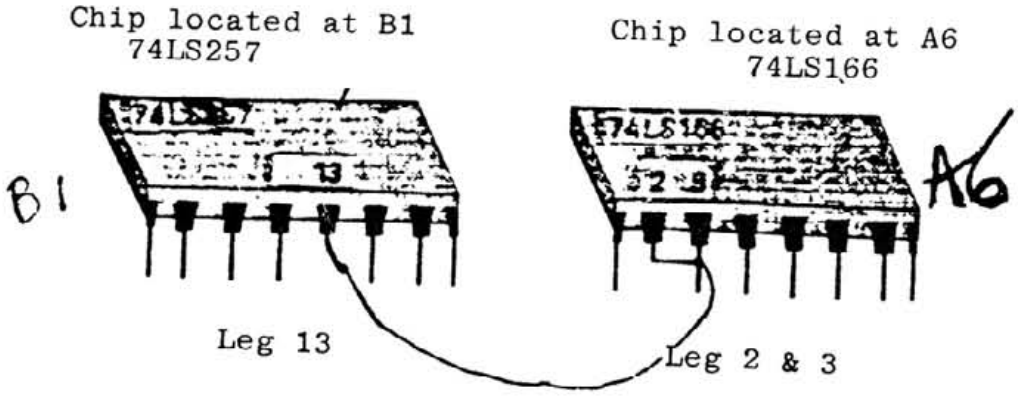
1. Locate chips B1 (74LS257) and A6 (74LS166), these are under the keyboard and will necessitate taking off the outer case of the ITT by removing the screws in the base plate.

2. Remove chips B1 and A6, or provide replacement chips if yoy wish the process to be easily reversible.

3. Carry out modification and linkage as shown in diagram and replace chips.
Note:-
Leg 13 of B1 is no longer located in the chip holder.
Leg 3 of A6 is still located in chip holder.

4. the effect of this modification, is to continue the colour adjacent to th tram line , through the tram line, so in essence, the tram lines are "smeared out", and the resultant effect is quite amazing.

5. for the "sophisticated" a hardware modification is now available (£35.00 excl VAT) which physically changes the ITT2020to give a 280 X 192 dot screen in Hi-res. Text and low-res graphics are unaffected.



APPLESOFT / PALSOFT ONERROR DIFFERENCES.

By Ian McKelvie

Previous articles have given details of some of the differences between APPLESOFT and PALSOFT. There is another important difference which I have not seen documented which may cause a program written in APPLESOFT to crash with an error message " RETURN WITHOUT GOSUB ".

This is caused by a difference in the way that the two languages handle the ON ERROR statement. Page 81 of the Applesoft Basic Programming Reference Manual warns that FOR...NEXT and GOSUB...RETURN causes problems in an ON ERROR routine as the pointers and the RETURN stacks are disturbed and suggests on Page 82 that a machine code routine is CALLed which will fix the problem. The PALSOFT version of the manual is a straight copy and is WRONG.

If such a program, with the fix, is run on an ITT 2020 with PALSOFT in ROM, then the problem will in occur. This is because PALSOFT has the fix in the PALSOFT ROM and the pointers and the RETURN stacks are disturbed again by CALLing the machine code routine necessary for Applesoft.

Examples that I have found are FILE CABINET in the Apple Contributed Software and in the C.C.A. Data Management System from Personal Software.

One way to overcome this is to eliminate from the program the machine code routine and the statements that CALL the routine. I have done this for both the above programs and it was a long process for the C.C.A. D.M.S. It is important not to remove the line number containing the CALL as this might be needed for a GOTO or GOSUB. I left them as a REM if there was only a single statement on the line. The other way I use is much quicker and involves using a language card and DOS 3.3.

I have modified the FP HELLO program to load APPLESOFT into the language card, instead of INTEGER BASIC, and to switch on the language card and write protect it. Thus any program running under DOS 3.3 in APPLESOFT will run without modification on my ITT 2020. The high res screens still display the vertical lines every 9th point and you must clear the screens before loading APPLESOFT as these lines may have been written to previously and cannot be cleared once APPLESOFT is running.If anyone has bought a copy of C.C.A. DMS and wants the ITT 2020 version, if they send me a disc with the APPLE version, I will make the necessary changes. The ITT version of File Cabinet is available from the Contributed Software Library.

A LETTER FROM ITT

Dear John,
Please accept my apologies for the delay in replying to your letters. However, I have today made arrangements for a Service Manual and circuit diagram to be forwarded to you under separate cover. I trust these will be of use to your members and should they wish to purchase further copies they are available from our Paddock Wood Service Dept., Eldon Way Trading Estate, Paddock Wood, Kent, RN12 6BE.

Please note that any servicing or technical queries relating to the ITT2020 should be referred to Mr Albert Grimwood at Basildon 3040, extension 108.

Yours sincerely,

K.A.Mace
Sales Manager- Micro Systems

*** Thank you Ken. We are sure there will be a number of calls for these from the library.
*** If you do want them, contact John Rodger. If you want to know how much they cost because you are too far down the list to wait:- the circuit diagram (a very nice poster in its own right) is £2.35(the order code is 24151-0-265;the service manual is £11.50 and order code is 24159-0-228 ; the disk drive service manual is £5 and order code is 24253-0-251.)

APPLE / ITT COLOUR CONVERSION CHART

APPLE	COLOUR	ITT
0	BLACK	0
1	MAGENTA	11
2	DARK BLUE	8
3	PURPLE	1
4	DARK GREEN	4
5	DARK GREY	10
6	MID BLUE	9
7	LIGHT BLUE	13
8	BROWN	2
9	ORANGE	7
10	LIGHT GREY	5
11	PINK	3
12	LIGHT GREEN	14
13	YELLOW	-
14	AQUA	12
15	WHITE	15
-	MID GREEN	6

HI-RES

0		BLACK		0
1	GREEN		BLUE	1
2	BLUE		GREEN	2
3		WHITE		3
4		BLACK		4
5	RED		AQUA	5
6	VIOLET		RED	6
7		WHITE		7

APPLE DISK & MEMORY UTILITY

- Repairs Blown Disks
- Reads Nibbles
- Maps Disk Space
- Searches Disks
- Searches Memory
- Edits Disk Sectors
- Outputs Screen to Printer
- Displays Memory In HEX/ASCII

THE INSPECTOR

These utilities enable the user to examine data both in the Apple's memory and on disks. Simple commands allow scanning through RAM and ROM memory as well as reading, displaying and changing data on disk.

Read and rewrite sections of Random Access files. Reconstruct a blown VTOC. Weed out unwanted control characters in CATALOG listings. UnDELETE deleted files or programs. Repair files that have erroneous data. All without being under program control and more.

You may transfer sectors between disks. This allows you to transfer DOS from one disk to another thereby saving a blown disk when all that's blown is DOS itself, or to restore a portion of a blown disk from its backup disk.

Its unique NIBBLE read routine provides a Hi Res graphical representation of the data on any track allowing you to immediately ascertain whether your disk is 13 sector or 16 sector. Get an I/O error - is it because you have the wrong DOS up? is it because of a bad address field? or a bad data field? or because a track was erased? This will allow you to tell in an instant without blowing away any program in memory.

The INSPECTOR even lets you search through an entire disk or through on board memory for the appearance of a string. Now you can easily add lower case to your programs (with LCA).

Do you want to add so called illegal line numbers into your program? or have several of the same line numbers in a program (like the professional programmers do)? or input unavailable commands (like HIMEM to Integer Basic)? or put quotation marks into PRINT statements? Here's the easy way to do them all!

AND MORE

The INSPECTOR provides a USER exit that will interface your own subroutines with those of the INSPECTOR itself. For example, just put a screen dump routine (sample included in documentation) at HEX 0300 and press CTRL Z. The contents of the screen page will print to your printer.

ROM RESIDENT ROUTINES

The INSPECTOR utilities come on an easily installed EPROM. This makes them always available for instant use. No need to load a disk and run a program.

FULLY DOCUMENTED

Unlike other software of its kind, The INSPECTOR comes with an EASY to understand manual and reference card. Examples and graphics help even the uninitiated use the power of these utilities. And furthermore, we offer the kind of personal service which you have never experienced from a software vendor before.

SYSTEM REQUIREMENTS

All Apple II configurations that have access to Integer Basic (either in ROM or RAM) will support The INSPECTOR. Just place the chip in empty socket D8 either on the mother board or in an Integer firmware card. Apple II+ systems with RAM expansion boards or language systems will receive the INSPECTOR on disk to merge and load with INTBASIC.

And if you have an Apple II - without either RAM or ROM access to Integer Basic, you will still be able to use The INSPECTOR because we are making available 16k RAM expansion boards at a very affordable price. Not only will you be able to use The INSPECTOR, but you will also have access to Integer Basic and other languages.

The INSPECTOR on DISK or ROM £35 Ram Extension Card RAMEX £95 Prices fully inclusive

MICROSOURCE 1, BRANCH ROAD, PARK STREET, ST.ALBANS.

sbd GAMES

Cribbage: Cribbage offers a totally challenging game. The board is done in Hi-Res Graphics and you can watch the scores being pegged. **£12.95**

Missile Defence: You command the Missile defence centre. As missiles drop from the sky you must shoot them while trying to protect your six cities and three missile bases. **£17.95**

Star Mines: The NEWEST arcade-style game. There are 5 levels of play. You must keep shooting and hitting the targets to achieve the next level before the clock times you out. **£14.95**

Hi-Res Soccer: The only real-time action SOCCER game for your Apple. All tackling, goal kicks, throw-ins, corners, are shown in full Hi-Res graphics. Fully animated players. **£22.95**

Alien Rain: The aliens swoop down from all sides in swift attack. Formerly sold as Apple Galaxian. **£12.95**

Snoggle: You are Snoggle, wending your way through a maze of ghosts who will eat you if they catch you. You must be quick and bold. Please note, the upper levels of this game are very difficult. **£12.95**

The Wizard & the Princess: In this adventure you must do battle against an evil wizard to save the life of the Princess. **£17.95**

Tarturian: As you explore 160 rooms you gather weapons and treasure for the final battle against the Tarturian. **£14.95**

Creature Venture: You have inherited your Uncle Stashbuck's mansion. You must rid it of the terrible creatures to find your uncle's buried treasure. **£14.95**

THE GALACTIC SAGA:
The Galactic Empire I – **£12.95**
The Galactic Trader II – **£12.95**
The Galactic Revolution III – **£12.95**
Tawala's Last Redoubt – **£14.95**

Add 15% VAT. Postage and Packing Free
Write or phone for full catalogue of available software.
Dealer enquiries welcome. Contact:

S.B.D. SOFTWARE
15, Jocelyn Road, Richmond TW9 2TJ.
Tel: 01-948 0461. Telex: 22861

JAILBREAK 2020

COME OUT FROM BEHIND THOSE HI-RES BARS

Now you can run any hi-res program written for your Apple on your 2020 witout the annoyance of the 40 vertical bars. Chess and complex games are a joy to behold.

* Saves having to modify programs to 2020 standard (no mean task with complex games which can be over 16K bytes long).

* No soldering required - Fit in five minutes by following the simple instructions.

* Switchable between Apple and 2020 nodes.

* Compatible with Apple PASCAL and Microsoft Softcard BASIC.

* Gives 280X192 hi-res dot screen like an Apple (the 2020 normally gives a 360X192 hi-res dot screen). Text and lo-res are unaffected.

* Does not support hi-res colour. Colours produce various shades of grey on a b&w display.

£33.95 inclusive.

MICROSOURCE

1, BRANCH ROAD, PARK STREET, ST.ALBANS.

TEL: 0727 72917 & 09273 75093

HEX/DEC COMPETITION RESULTS

The winner in the Hexadecimal to Decimal class of the competition to find the shortest Applesoft routine was Ray Harris. Here is his entry, taking 71 bytes :-

```
10 INPUT "HEX ?"; H$: FOR A = 1 TO LEN (H$) :
N = ASC ( MID$ (H$, A, 1) ) : N = N - 48 - 7 * ( N >
64 ) : D = D * 16 + N : NEXT : PRINT D
```

In the Decimal to Hexadecimal class, John Rodger and Ian Trackman submitted virtually identical winning entries. A number of competitors, including John and Ian, provided for their routines to accept negative decimal numbers, so making them more useful. Here is John Rodger's 88-byte routine :-

```
10 INPUT "DEC ? "; N : N = N + 65536 * ( N < 0 ) :
A% = N / 256 : POKE 69, A% : POKE 70, N - A% *
256 : POKE 58, 65 : POKE 59, 249 : PRINT "$"; :
CALL -327
```

Ian Trackman also sent in the shortest (96 byte) Decimal / Hex entry which did not use machine code calls :-

```
10 INPUT "DEC ? ";N : N = N + 65536 * (N < 0 )

20 R = INT ( N / 16 ) : H = N - R * 16 : A$ = CHR$
( H + 48 + 7 * ( H > 9 ) ) + A$ : N = R : IF N THEN
20

30 PRINT "$"A$
```

Unfortunately, a few entries had to be disqualified because they involved going down into the Monitor in order to enter a machine-code routine and this meant that the entire routine was not written in Applesoft.

We also have to commiserate with one of our losers, whom, to spare his blushes, we'll call "Ivor Webb", and who promised to eat his Apple if anyone found a shorter solution than his ! (He's lucky we're not a PET user group.)

The machine-code calling method listed above demonstrates a very useful way of calling up from Basic those Monitor and ROM routines which have to be entered with the 6502 A, X and Y registers set in a particular way. Here is a "line-by-line" version of Ian Trackman's routine, this time with his explanation:-

```
10 INPUT "DEC ? "; N
20 D = 256
30 N = N + D * D ( N < 0 )
40 H = N / D
50 POKE 69, H
60 POKE 70, N - INT ( H ) * D
70 POKE 58, 65
80 POKE 59, 249
90 PRINT "$";
100 CALL -327
```

Lines 50 / 60 - Memory locations 69 - 71 ($45 - $47) are used by the Monitor as temporary storage for the contents of the A, X and Y registers. (Locations 72 and 73 hold the processor status and stack pointer). Here, we Poke the decimal number modulo 16 into the A register and the remainder into the X register.

Lines 70 / 80 - Locations 58 and 59 ($3A and $3B) are used by the Monitor to store the "Program Counter", that is the address in memory where the program will continue after a Monitor List, Go, Stop or Trace command. In this case, we Poke it (in the usual 6502 low / high order) with the decimal equivalent of $F941. $F941 is the start of a routine in the Monitor (called PRNTAX) which prints out the values (in Hex) of the A and X registers.

Line 100 calls the Monitor routine at $FEB9 which puts the values stored in $45 - $48 into their correct registers and then jumps to the routine whose address is in locations $3A and $3B.

This method can be used for calling up a variety of useful machine-code sub-routines including very fast screen switching and memory moves.

Ray, John, and Ian each win two discs of their choice from the Software Library.

This month's competition consists simply of - Setting a competition for the next issue! The winner will as usual receive two free discs, in addition to the honour of judging his competition and writing a commentary on the results.

A WARGAMER'S VIEW OF MICROCOMPUTER GAMES

-CONCLUDED-

By Robin Hood

COMPUTER AMBUSH

Once again we have a box to contend with, only this time the box is even bigger! It's about the same size as a standard SPI game box, which is totally unnecessary for what it contains. It contains one rule book (with 6 pages of rules and no less than 8 pages of dossiers on the soldiers that can be involved in the game), two plastic coated mapboards, two squad cards, a game selection card, two chinagraph pencils and one 5.25" floppy disk.

Again, as with Bismark, there are two versions available. One for the APPLE and one for the TRS-80, or so I am told to believe. (Available on disk only).

Ambush is a simulation of squad vs squad combat in France during 1944, or to be more precise, a single level, rubble strewn, French village. The game is nothing more or less than a very basic form of SPI's Sniper.

On one side of the squad card there are two charts listing the various abilities of the 10 Americans and the 10 Germans. These abilities range from 1 to 10 (1 being the worst and 10 being the best). The abilities themselves cover the weight, strength, intelligance, dexterity, power of observation, throwing, firing and hand-to-hand combat. One or more of these may factor into whatever action a soldier may be undertaking during his move(s) in any one turn. On the reverse side of this card there are two order summary charts, one for movement and one for combat/weapons.

The game selection card offers a choice of no less than seven scenarios. The first three are NCO Training, Ambush and Raid. They are played against Otto Von Computer, the first one being nothing more than a short introductory game for new players. In the second one the American has to set up an ambush of a German patrol (which can start in four different places on the map and with a variable number of men). The third is a raid on the government building at the northern end of the map, where the American must place a charge of plastic explosive in a specific square inside said building.

The two-player scenarios are Ambush, Strongpoint, Patrol and Free Form. In the Ambush the German sets up an ambush of an American squad. In the Strongpoint the American is defending the factory in the southwest corner of the map against a German assault. The Patrol is a meeting engagement and speaks for itself, as does the Free Form. You could even try for computer non-violence and run relay races!!!

The plastic coated mapboard has a square grid superimposed over it to regulate movement, though the grid does not (thankfully in this case), show up on the screen map. The xx co-ordinate (column) 11 to 48 cross-indexed with the yy co-ordinate (row) 11 to 64 pinpoints the location of the men for fire and movement. On the back of the map sheet there is a map key, a terrain chart and a listing of the buildings; from a factory, farm, school, government building and church to houses and shops. There is even a statue!

Once the game is up and running you will be greeted by the usual hi-res display of the company name, which in turn is soon followed by a graphical display of the box-cover. In fact it's 10 times better than the box artwork. After a short pause you will be asked if you want a (Q)uiet game or one with (S)ound effects. Whether you wish to play against the (C)omputer or a (H)uman opponent, and in response to C or H you will be asked to select a scenario. You now find yourself in the scenario set up mode and you will be asked to input answers to yet more questions; not necessarily in this order:- Do you want a (B)lind game, where an enemy soldier will only be seen if in the line of sight of one or more friendly men, or do you want an (U)nlimited one in which all soldiers are displayed. Do you wish to (K)eep or (E)liminate any of your men. Input of a five letter codeword, and so on... then you will come to the most crucial question. How many time points do you wish to allocate to the turn?!

You can type in a number between 1 and 99 (anything less than 10 is a waste of time!) but if you are playing with more than half a dozen men per side and you type in 99 the computer execution phase could last an hour or more in real time terms! Two men take about 3.5 minutes of 99 time points. Of course the computer guess-estimates this to a certain extent because the actions of the men (or lack of them), has some bearing on the time countdown. Generally speaking, the more men you have in play the longer the execution phase. As casualties are taken the execution phase speeds up so more time points can be added as the game progresses. As I have discovered, if you have too many men doing too much with too many time points you run the risk of blowing the memory, which of course means a re-start!

With the set up completed the American player is called to the screen by a little tune (the first few bars of Yankee Doodle would you believe!), and upon the correct entry of his codeword he will be asked if he would like the see the (M)ap, give (O)rders, see the (S)tatus of one of his men or (F)inish his orders for the turn. You can jump from the map to the status display and back again at will, giving orders to your men in either mode.

The status display shows the name and rank of the soldier, his wound level (0 to 99), his command control rating, his xxyy location, his current energy level and endurance, whether he is (U)pright or (P)rone and of course his weapon type. In fact everything you need to know about the current state of each individual is listed in his status report.

Such things as:-

Wounds (+50 he is in shock, prone and out of play. 99 he is dead).

Command control (.1 to 1 (the lower the number the less likely he is to obey orders)).

Weapon ((R)ifle. (A)uto rifle/submachinegun/B.A.R. (M)achinegun).

Is it loaded, in use (RYN would mean rifle/in use/unloaded. ANY would mean auto rifle not in use/loaded!).
The number of grenades carried, a knife (which would become a bayonet if rifle armed), even a garotte is catered for! In addition to that there is also a listing of his current orders, if any, and how many time points the soldier has left in order for him to complete the first set of these outstanding orders.

You can give a soldier as many orders as you like, until the time points for than turn reach or drop below zero. Once a soldier has gone into minus time points you cannot give him any more orders, though you are free to change his orders and give him new ones as many times as you like until you are happy with the actions he has to perform for that turn. Of course there is no guarantee that he will complete all the orders given to him, or indeed obey any of them!! Energy and time is taken away for each action, but the amount of both depends on the individual soldier. Cheng cannot throw a grenade or fire a rifle to save his life, but he is a wizz with his satchel charge.

Resting. Provided a soldier does not end the current turn with negative time points he will regain a certain amount of lost energy and endurance. Again this will depend on the individual soldier and his strength rating. The stronger he is the more he will be able to do before becoming fatigued. In addition to this a soldier will also lose or gain a command control point each turn. This depends on certain actions and his location in relationship to his squad leader. The squad leader being the highest ranked soldier in play who is still active.

The map is displayed in hi-res graphics, or rather a third of it is. It is too big to fit onto the screen in one go! This may seem to be a bit of a bother at first, but once you get into it the commands to change the map display become second nature. All you have to do is type (M) for map and then any number between 11 and 47. The number you type becomes the bottom line or edge of the map. Apart from the terrain, buildings, hedges, walls, windows, doorways etc, etc., the American will see his men depicted as the first letter of his last name and the Germans as Swastikas. The German will see his men as the first letter of his last name and the Americans as strange looking squashed up Stars & Stripes. Enemy in shock are seen as a simple cross, while enemy dead are seen as Skull & Crossbones!

Although you can give orders to your men while looking at the status display and the map sheets provided, I prefer to deal with the men and their orders while looking at the map on the screen as you can actually see the men move into their projected new locations (there are also a couple of small printing errors on the map sheets- not that they matter that much!!). If you do not feel happy with the location of one or more of your men it is a simple matter to change his/their orders. In fact giving orders is a very simple matter, though at first glance it looks far from simple.

After keying (O) for orders you are asked to type in the first letter of the last name of the soldier in question. Having done that the name of the soldier, his current energy level and the number of time points allocated to that turn appear at the bottom of the screen. New orders will cancel out any outstanding orders!

Movement orders consist of two letters, two numbers and another letter. If you were to input anything else a rude noise would be emitted from the sound box to tell you that whatever it was that you just tried to do was illegal! (in fact this would be your reward for any kind of mistake in response to any question). The first letter is always (M) for movement, the second is always an (R) or an (I) for regular or irregular movement. The first number can be anything from 0 to 8 (0 is a 360 degree scan; which is illegal if the 2nd number is not 0. 1 is north, 2 northeast and an 8 is northwest), to indicate a direction for movement (a 9 here would mean that you wish the soldier to engage in hand-to-hand combat!). The second number can be anything from 0 to 9 (which is the number of squares you wish the soldier to move). The last letter represents the type of move that is to be made, from (W)alk, (R)un, (C)rawl, (D)odge, (F)all down to (S)tand up. Once again the amount of time and energy lost for each action depends on the individual. (Try crawling more than one square....and see how far you get!!). A movement order might look something like this; (MR62D) or (MI38R). In the first example you would be moving regular southwest for 2 squares while dodging. In the second example you would be moving irregularly east for 8 squares at a run. Regular movement means that you end up facing in the direction of your move. In the above example it would be southwest. If you move irregularly you end up facing in the direction you were facing before you began the move. In other words, if our friend in the second example began the move facing in direction 7 (west), he would at the end of his run east turn and face west again. Irregular movement and any kind of diagonal movement costs twice the number of time and energy points.

As I have already mentioned there are three types of fire weapons. There are two types for fire combat. FSxxyy is an order to fire a single burst at a target in a specified location. FApptt is area fire, or opportunity fire by any other name. FA being the fire order and pp being the number of percentage points that you wish to allocate to the fire. 01 means that you will fire at the merest suggestion of a shadow while 99 means that you will only fire when you see the whites of the enemies eyes (and in some cases you wont fire at all!). And the final part of the command; tt. This being the number of time points you have allotted to the soldier to carry out these orders before he goes on to do something else, if anything. 10 is the minimum number of time points that all soldiers must spend in order to carry out this action.

Then there are orders covering the preparation of grenades, knives, bayonets, reloading etc, etc. Most of them are two letter commands.

The German player (if there is one), is called to the screen by the first few bars of an unidentifiable tune once the American has (F)inished his order phase. When the German has done his bit the disk goes for a spin, or three, and then the countdown phase begins. The most time consuming part of the game as the time points that you have allocated to that turn begin to drop before your eyes! During this phase you will hear pops, clicks and other strange noises, if you have asked for a game with sound effects that is.

When the countdown is over the American will hear Yankee Doodle calling him back to the screen (once the disk has stopped it's second round of three spins!), and upon the correct entry of his codeword he will be asked if he wishes to see the (A)ction, see the (S)tatus of one of his men or (F)inish his report phase. The first thing to do is see the action! Now you will be able to see what you may have heard during the countdown - gunfire, explosions and men screaming as they are hit! After watching the action, as many times as you like, it is advisable to go through the status reports of all your men, especially if they have been fired upon. Once the German player has finished with his report phase the players will be asked if they wish to (C)ontinue, and if so, how many time points they wish to play/allocate to the next turn. On the other hand you are given the option to (S)ave the game on one of your own blank disks and come back to it at some other time.

This is a very good game, but there is room for improvement. If you know and are prepared to accept the limits of the program you can have hours of fun. Apart from the long waiting periods during the countdown phase, the major bug in the game seems to be the limitations on the number of men that you can deploy. I have tried several games with 15 plus men and in all cases one of two things happen. Either the computer blows it's memory after counting down the time points for the first turn, or if it gets past that snag, the Germans stop moving after a few turns and stand there picking their noses (or whatever it is they do when the computer has lost all interest in them!). Then, if you like, you can always pick the nits! You cannot move diagonally through apertures (doors and windows). It costs more time points to fall down than it does to stand up! And you are stuck with a bunch of soldiers who's equipment cannot be changed or exchanged from one soldier to another during a game, or from game to game. Still, despite these few minor points it is an excellent attempt to cover a very complicated subject and it shows great promise for the future. It is definitely a move in the right direction as far as this wargamer is concerned.

I am right in the middle of writing this article when I learn that S.S.I. have just released three more titles. One is called Apocalypse Then, which is something to do with para's in action in Normandy if I have got my information correct. Then there is Torpedo Run and an SF game (title as yet unknown!).

Finally backtracking to my preamble, there are some games that defy classification. Take Three Mile Island (from Muse), and The Prisoner (from Edu-Ware). I have received some favourable reports on the former; a game in which you try to keep the lid on a nuclear reactor! As for The Prisoner which is based on the old ATV series, this is an adventure with a difference, and boy what a difference. A very personal game that must be played solitaire so that you can keep any and all information on how to 'play' the game to yourself (never mind the final objective, which is of course to escape!). No help, no hints and no tips on The Prisoner..... find out for yourself!

SPECIAL INTEREST GROUP FOR THE HANDICAPPED.

By Glyn Vernon

Being handicapped very often means being bored, and the more severely handicapped you are the more bored you are likely to be, particularly if the handicap is a purely physical one coupled with a normal or above average intelligence.

You start life by being handicapped educationally, partly through lack of ambition on the part of most of those in charge of your education, but mainly because you very often cannot use the tools other people take for granted. Simple things like turning the pages of a book might be beyond your physical capabilities; or writing might prove an insurmountable problem. You might be too deaf to hear what is being said to you. Or too blind to see. Or indeed any combination of these and many more possible disabilities, too numerous to mention here.

Almost everything in later life stems from your standard of education. This is true for everyone but the handicapped person finds himself landed with a double disability. There is the obvious one caused by his physical shortcomings; and a second one caused by the poor quality of his education. These two then combine to produce a social disability which stems from both without being directly related to either. The disabled person, in spite of normal or above normal intelligence is thus typically disadvantaged both socially and in employment opportunities. He is condemned to a life at home; a life of waiting. Usually waiting for nothing.

In such circumstances anything which means freedom in any sense of that word is of paramont importance. Recent years have seen a great interest in appolying new microprocessor technology towards helping the disabled, and although much of the early work was centred around the PET, most people working in the field have now turned to the APPLE. The APPLE has greater flexibility; it can be interfaced with most things as well as being simple to operate. And most of all it brings liberation, becoming at the same time an aid, a source of education, and a stimulating potential source of income.

This liberation does indeed follow the three levels mentioned above. Many severely disabled people have communications problems, often caused by speech defects. For these, if nothing else, the APPLE or perhaps some cheaper micro, can at least provide a very efficient communications aid. In many cases it will

amount to no more than this, simply typing words on the keyboard that appear on the monitor. A simple program can be written to enable this to be achieved without panning, or annoying messages like "syntax error" appearing every time the return key is depressed; and that is all that particular micro is used for because the user is incapable of, or doesn't want to use it in any other way. You might think this a waste of the potential of the machine but just think how liberating it is just to be able to express oneself clearly. And some of the specialist aids on the market which are designed to do just this job, cost many times the price of the cheapest micro.

The second level of user may wish to go beyond this most simple application, and to use the machine for entertainment purposes, such as playing games. This person has more ability, or greater interest than the first type, but still doesn't want to program the machine. And the third type will want to go the whole way, either to create programs for his own pleasure and use, or to learn to program commercially, thus opening up employment opportunities denied to him in the past.

Among the charities working with the handicapped, two have led the field in applying the micro-processor to the needs of the handicapped. AIDIS Trust is a fund specifically set up to provide such aids when needed, and the Spastics Society has done much work in opening up the field, notably by setting up the Professional Workshop in Neath Hill, Milton Keynes. The Workshop has a double function; firstly to provide twelve severely disabled people with an environment in which they can earn a living doing professional work, mainly in the computer field, and secondly to use the expertise within the project as a centre of knowledge for APPLE users and those helping them.

The co-ordinator of the project, Peter Deakin, and those partners (i.e. the disabled participants) directly concerned in using the APPLE are members of the St. Albans branch of the British Apple Systems User Group and because of this the group has fathered the idea of setting up a Special Interest Group to further the use of the APPLE (and indeed the micro in general) to help the handicapped user. We need all kinds of help, mainly in small, seemingly trivial ways, which are nevertheless of major importance to the user.

For example it is easy for those of us familiar with micros, knowing how simple they are to set up and operate, to forget that the world is not yet populated by computer-literate people.

Mention the word computer to most people and they still tend to think they are being led into a mysterious technology which is forever going to be beyond them. Yet we are talking of dumping a computer, perhaps on the parents of a disabled schoolchild, and setting the system up and leaving them to it. At the moment the servicing of these systems is largely carried out by the Workshop. Anyone willing to be on the end of a telephone to help such a user living in the vicinity would be very useful. Remember we are not asking anyone to undertake extensive repairs of broken computer systems; most of the problems are simple ones. A card coming loose because a system was moved, for instance, or problems caused by lack of knowledge. Recently, a father desperately grappling with a new technology in order to help his son, crashed the system by putting the printer card in slot 0. It was the first slot, and according to his conventional logic, the first slot should be slot 1. A simple mistake but enough to crash the system for days, until someone travelled a fairly long distance - to spend two minutes rectifying the error.

Another, more experienced user might simply need someone he can talk to about his machine. We have all experienced getting bogged down in a program and how helpful it can be just to talk the program over for a few minutes with another programmer. And there are other ways people could help too.

Programs written for use by the non-disabled user often need a certain amount of re-writing in order to be easily used by the handicapped. Or the system itself might need minor adaptations, perhaps to provide locks for the shift or control keys,for those who cannot depress two keys at once. Programs written for other computers, such as the PET might need re-writing so they can be used on the APPLE. Or perhaps someone simply knows of a disabled person they feel could benefit from a system.

Whatever ways people could help it is felt that a special group, meeting perhaps only three or four times per year couldhelp co-ordinate progress and stimulate new ideas. Accordingly it has been decided to initiate such a group and the first meeting will be held at a date to be announced. Anyone interested in joining the group, or who simply feels that he would like to help in other ways should contact BASUG.

It is impossible to over-estimate the sense of liberation the micro offers the disabled person. Anything that can be done to further the use of the machine in this field just has to be well worth the effort.

MINUTES OF THE FIRST AGM OF BASUG HELD AT THE SPASTICS SOCIETY HEADQUARTERS 12 PARK CRESCENT ,ON 21 JUNE 1981.

The meeting was opened at 2.45 pm. by the Chairman/Secretary John Sharp.

There were 30 members present in person and 16 proxies held by John Sharp.

1. The Acting Secretary to read the notice convening the meeting.
After thanking the Spastics Society for the use of their premises, and Peter Deakin for making the arrangements with them the Chairman/Secretary read the Agenda.

Before moving on to item 2 the Chairman gave a summary of the first six months of the club. This is summarised as follows:-

The membership was now about 350, not only nationally in Britain, but stretching through Europe and Africa and Hong Kong, which is quite an achievement in just 6 months.

This has meant a great deal of work and some objectives such as SIGS had not yet got off the ground. On the positive side:-
(i) The group is affiliated to the IAC so we have a voice, albeit a small one, in the Apple world at large.
(ii) Software was a thriving concern in both libraries.
(iii) The magazine would be a credit to any publishing company, never mind an amateur group.
(iv) The bulk buying and selling of disks, etc, has saved many members their subscription many times over.
(v) The exchange of information was increasing at an exponential rate.

The amount of work is considerable as the letter sent to members outlines, and this has meant an intrusion into the normal life of the Committee as well as hindering their programming. Regretfully Martin Perry had to stand down because he could not fit the time in with his job. I would like to thank him and everyone else who has helped, not only on the Committee but also in copying software and numerous other jobs.
We as a Committee feel we have got it right, but there is a need to rely on more than purely voluntary support. Certain proposals had been made in the letter to members and the Constitution so that the Group could carry on in a similar vein. Otherwise the group would have to take a few steps backward to a more amateurish level, with a drastic cut down in HARDCORE and the other facilities of the club.

2. To appoint I.L.Trackman as chairman of the meeting for the purposes of the adoption of the Constitution and elect a Chairman of the group.

The resolution that this motion be carried was put by David Bolton and carried by the meeting whereupon Ian Trackman took the chair. Discussion on the proposed Constitution sent to members with Hardcore No. 3 then commenced.

There were no queries or points to be discussed under items 1-4 of the Constitution.

There was much discussion on Clause 5 regarding the termination of membership. The meeting thought an expelled member had the right to some form of appeal. The Clause finally agreed, proposed by Vernon Quaintance and amended by Ted Lepley was :-

5(a) the Committee is empowered to terminate the membership of any member whose conduct, in the considered opinion of the Committee, renders him unfit for membership of the Club.

(b) Before any member's membership is so terminated, the matter is to decided in accordance with the rules of natural justice.

(c) The expelled member shall have the right to have the matter considered by an SGM convened in accordance with Clause 10.

The point was made that an expelled member, not being a member, was not entitled to attend such a meeting. It was pointed out that the first resolution of such a meeting would be to give him right of audience.

The next Clause to generate discussion was 6(c).
It was decided that there was a problem with the Clause as it stood since the situation could arise in which a post could be unfilled because there was no nomination in the time allowed. It was proposed by Kevin Brown that nominations be allowed at the AGM. The proposal was amended by Ted Lepley to read :-
In the event of there being an insufficient number of nominations, nominations may be submitted at the GM.
There were discussions as to whether there should be a qualification on length of membership before a member could stand for the Committee. There was no proposal on this discussion.

Clauses 6(d) and 6(f) provoked discussion as to how such facts should be communicated to the members. A new Clause 6(g) was proposed by Ted Lepley:-
6(g) members shall be notified of any cooption under Clause 6(d), or change of rules under 6(f) by the next general written communication to them.
Clause 7 led to a number of questions.

The reasons for such a Clause were given by the outgoing Committee as (i) a means of paying those working for the Club, with the possibility of paying for outside help and (ii) a means of separating the commercial aspects of the club for tax purposes etc.

It was pointed out this Clause did no more than empower such an entity to be set up. A special meeting would be required to do so which would require a simple majority if passed in the Constitution as proposed. If it were not passed as part of the Constitution then an amendment would be necessary which would require a two thirds majority under Clause 11(e).

John Sanderson thought Clause 8 (a) put the financial year end too far from the AGM to present a balanced view of the financial state of the club. Other members pointed out that it was reasonable and no proposal was made.

The last sentence of Clause 8(b) was proposed to be amended by Kevin Brown as follows:-
All cheques and other instructions given to any such bank or relating to investments are to be signed by (i) one of the Chairman, Secretary or the Treasurer and (ii) one of those Committee members designated for the purpose by the Committee.

In Clause 8(c) Vernon Quaintance moved that the second 'produced' be removed as redundant.

It was proposed by David Bolton that the opening part of Clause 9 be amended to read :-

"An annual general meeting shall be held once in every calendar year not later than 31 July in order to transact the following business:-"

It was also pointed out that no provision was made to have a report from the Committee nor to approve the minutes of the previous GM. Consequently two amendments to Clause 9 were proposed,

9(a) To approve the minutes of the previous General meeting.

9(b) To receive the annual report of the Committee.

When Clause 10 came up for discussion the following points were made. In the event of a single member or a small group of members wishing to call a special meeting the problem might arise that they might not know 20 members, by the very fact of the club covering such a wide geographical area. In view of this it was proposed by Vernon Quaintance that the following addition be made:-

" A member seeking support for the calling of a Special General Meeting may require the Committee to circulate the proposed resolution with the next written communication to members."

To avoid ambiguities in Clause 11 the following amendments were proposed:-

"11(d) Twenty members in person or one quarter of all members in person or by proxy, whichever is the less, shall form a quorum."

"11(e) All resolutions require a simple majority of votes of members present in person or by proxy at the meeting, except for resolutions to amend the Constitution or to wind up the Club, which require not less than two thirds of the votes cast in favour in order to be passed."

These points having been discussed the amendments were voted on and all passed.

3. Adoption of the Constitution

David Bolton moved that the proposed Constitution be adopted. This was seconded by Ted Lepley. There being no objections or abstentions the amended Constitution was declared adopted by the Chairman.

4. Election of the Committee

The Chairman then moved on to the election of the Chairman of the club for 1981-82.

It was proposed by John Sharp,seconded David Bolton that Frank Kay be elected to the post of Chairman. On his nomination form Frank Kay had declared his willingness to stand and that he had a commercial interest in the Apple in that he is an Apple Dealer.

There being no objections or abstentions he was declared elected.

Since Frank Kay was unable to be present, Ian Trackman stayed in the Chair to elect the Secretary.

It was proposed by David Bolton, seconded John Rogers, that John Sharp be elected to the post of Secretary. On his nomination form John Sharp declared his willingness to stand and declared he had no commercial interest in the Apple.

There being no objections or abstentions he was declared elected.

Since there was an elected member of the Committee present, Ian Trackman handed the Chair to John Sharp.

It was proposed by John Sharp and seconded by John Rogers that David Bolton be elected to the post of Treasurer. In declaring his willingness to stand, he also declared that his commercial interest in the Apple was as a software author and distributor.

There being no objections or abstentions he was declared elected.

The following four members of the Committee were also elected with no ojections or abstentions:-

Tony Williams, as Membership Secretary, proposed by John Rogers and seconded by David Bolton. He gave his commercial interest in the Apple as a software author /distributor.

John Rogers (as Software Librarian),proposed by Tony Williams and seconded by John Sharp, declared no commercial interest in the Apple.

Edward Payne (as events organiser) was proposed by John Sharp and seconded by Tony Williams. He declared that he is a freelance at the moment but will be associated with an Apple dealer in the near future.

John Rodger (as information officer), proposed by David Bolton and seconded by Bob Raikes declared his commercial interest as a software author.

5.Subscription rates for the year 1982

It was proposed by the Treasurer that a decision on the rates be deferred until later in the year since the group was growing so rapidly that no planning could be made so far ahead. This was then agreed in discussion and the following motion put forward by Ian Trackman was seconded by David Bolton.

"The subscription rate shall stay the same subject to any later SGM."

This was agreed by the meeting and declared passed.

6.AOB

(i) John Sanderson proposed a vote of thanks to the outgoing Committee and all the helpers for their work over the first 6 months.

(ii) It was decided there was a need for a talk or other meeting at the next AGM in order to attract more members. It was suggested that a longer session perhaps a full day or a weekend conference would be even more in order. The Committee agreed to look into this.

There being no other business the meeting was declared closed at 5.35 pm.

PRINTER INTERFACING - PART 3

By Chris Murphy

In the final article of this series, I shall be considering the Apple Communications Card and how this may be used to drive printers.

The Communications Card is, strictly speaking, not designed to be used as a printer interface. It provides the ability to communicate data serially in either full or half duplex mode at 110 or 300 baud, and normally would be used to enable the Apple to communicate with another computer. It is quite feasible to use the Apple as a terminal to a larger computer; the Appletel interface (which is similar to the communications card) allows the Apple access to Prestel, with the advantage using Apple as against a normal Prestel receiver of having a full Alpha keyboard and floppy disk on which to store Prestel pages. Another increasingly common use for the communications card is to enable Pascal users to drive their Apple from an external VDU terminal with the benefits of a larger screen. However, as the communications card conforms to RS232C standards (which were covered in the last article), it may be used to send data to a printer.

The communications card supports four pins on the RS232C interface:- transmit data, receive data, signal ground and data terminal ready (the minimum signalling required by a modem). Data sent via it is treated as raw data; therefore, if you are driving a printer, you need software driver routines to specify the output column width, and to supply a line feed after a carriage return. Also, as the card will output at a maximum of 300 baud, it is not intended for fast printers. The manual supplied with the card gives details of how to modify the card for higher baud rates; however, one of these modifications is not correctly specified, and if you require higher speeds, it may well pay you to specify this when ordering the card, as these modifications can be undertaken by Apple Level II service centres without voiding the warranty - if you modify the card yourself, you will void the warranty.

Having said this, some printers which have a degree of intelligence, can be driven directly from the Apple via a communications card without the need for a driver routine, especially if supplying a line feed after a carriage return is a switchable option on the printer itself. However, be prepared for some strange screen displays, especially when using Applewriter!

The following routine is a driver which I have used, and know to work. It assumes that you are using Applesoft and either DOS 3.2 or DOS 3.3. In order to load the routine, first boot up from disk, and then enter Monitor via CALL-155. Then enter the following:-

```
305:18 B0 38 98 48 AC 00 03
:08 A9 05 8D 03 03 A9 C0
:18 6D 00 03 8D 04 03 A9
:FF 99 78 07 A9 11 99 F8
:07 98 0A 0A 0A 0A A8 A9
:03 99 8E C0 A9 11 99 8E
:C0 A9 84 8D 01 03 A9 36
:85 2A A0 00 8C 02 03 84
:2B 28 90 0E AD E7 03 E9
:6E 85 2A AD E8 03 E9 00
:85 2B A9 63 91 2A C8 A9
:03 91 2A 68 A8 60 48 48
:AD 02 03 C5 24 68 B0 03
:48 A9 A0 2C 62 03 F0 03
:EE 02 03 08 20 A9 03 28
:68 48 90 E4 49 0D 0A D0
:0D 8D 02 03 A9 8A 20 A9
:03 A9 58 20 A8 FC A9 00
:85 24 AD 02 03 F0 09 ED
:01 03 E9 F7 90 04 69 1F
:85 24 68 60 6C 03 03 FF
```

Having entered all this, return to Basic by typing 3D0G. You can now save the driver routine onto disk. Assuming that you want to call it DRIVER, type BSAVE DRIVER,A$300,L$AF.

To use the routine, you must first BLOAD it from disk. Then you must POKE 768 with the slot number of the communications card, and POKE 769 with the output column width. Then CALL 775 will activate the driver routine. A normal PR#0 will return control to the screen. Thus to use the driver under program control, you would use something along the following lines:-

```
10 D$=CHR$(4):PRINT D$;"BLOAD DRIVER"
20 REM Get the driver into memory
30 POKE 768,1
40 REM Card is in Slot 1
50 POKE 769,132
60 REM We require 132 column wide output
70 CALL 775
80 REM Activate Driver
90 PRINT "THIS WILL NOW APPEAR ON THE PRINTER (IF YOU HAVE
REMEMBERED TO SWITCH IT ON!)"
100 REM Output text to printer
110 PRINT D$;"PR#0"
120 REM Return output control to Screen
```

-+-

This concludes the current series of articles, which I hope has proved of some interest and has at least demystified some of the terminology. I have had one or two queries already regarding problems with different printers, and will endeavour to deal with these as time permits, and to include the queries and replies in the magazine.

C/WP

C/WP COMPUTERS 108 Rochester Row, London SW1. 01-828-3127

Forecasting and management information specialists

Authorised APPLE dealers and level 1 service centre

	Typical RSP	C/WP Price	VAT	TOTAL
APPLE PROCESSOR				
Apple 48K Europlus	812	599	89.95	688.95
EXPANSION OPTIONS				
48K to 64K RAM card	-	75	11.25	86.25
Pascal 64K system	246	226	33.90	259.90
6809 system	-	240	36.90	276.90
Z80 CPM system	200	185	27.75	212.75
Real time clock	-	90	13.50	103.50
Videx 80 column system	220	200	3.00	203.00
DISC DRIVES				
1 Apple (143K)	397	380	57.00	437.00
2 Apple (286K)	708	690	10.55	700.55
2 Teac (286K)	-	550	82.50	632.50
2 Teac (655K)	-	840	126.00	966.00
DOS 3.3 upgrade kit	36	36	5.40	41.40
MONITORS				
Green 12"	196	190	28.50	218.50
RGB 8 colour 14" (not a converted television)	395	380	57.00	437.00
PRINTERS				
Silentype + interface	349	299	44.85	343.85
Epson 100	575	525	78.75	603.75
Paper Tiger 560	895	850	127.50	977.50
Qume Sprint 5	-	1450	217.50	1667.50
Diablo 630	-	1650	247.50	1897.50
Printer interface	-	100	15.00	115.00
SOFTWARE				
Visicalc 3.3/3.2	111	80	12.00	92.00
Visidex 3.3	111	90	13.50	103.50
Visitrend + Visiplot 3.3	144	125	18.75	143.75
PASCAL SOFTWARE				
Micromodeller	425	325	48.75	373.75
Decision Modeller	535	435	65.25	500.25
Microfinesse	595	495	74.25	569.25
Z80 SOFTWARE				
Wordstar mail merge	350	300	45.00	345.00
CIS Cobol	475	375	56.25	431.25
655K SOFTWARE				
TAM Accounting	-	600	90.00	690.00
ACCESSORIES				
10 discs	31	20	3.00	23.00
Library box	2	2	.25	2.25
2000 sheets paper	-	20	3.00	23.00
Thermal paper	28	28	4.20	32.20
Installation London area	-	200	30.00	230.00
Installation rest England	-	300	45.00	345.00

Delivery by Data Express free for cash with order, alternatively collect from C/WP with banker's draft. One year's warranty is standard with all hardware including daisywheel printers. Typical RSP's refer to prices advertised by manufacturers and not to prices previously charged by C/WP.

Accutrack Disks . . .

Because data reliability is the important difference in disk construction.

Anatomy of a disk.

Flexible disks are simple information storage devices consisting of a magnetic disk enclosed in a semi-stiff protective jacket. The disk rotates within the jacket while magnetic recording heads on your data or word processing systems "read" or "write" information on the disk's magnetic surface. Since disk operation is simple, it's relatively easy to make one that works. But building in reliability is something else again. It takes specialized technology to build disks that operate flawlessly over an extended period of time.

What counts in disk construction.

Key design objectives for a disk are listed below. How well a disk measures up to these objectives relates directly to the throughput, accuracy and overall costs for your data or word processing system. No disk measures up better than Accutrack.

- The magnetic coating must be precisely formulated and uniformly applied. Imperfections as small as five millionths of an inch cause signal dropouts, data checks and wasted processing time as well as errors.
- The disk surface must be absolutely clean, totally flat and permanently lubricated to prevent excessive head wear with subsequent signal degradation and eventual loss of information. (This is the most critical objective and the one that's most often compromised since poor operating results take a while to show up. It's also the area that most effects the long term reliability of your data.)
- The disk must be free to rotate within its jacket without internal drag to avoid further data checks, excessive processing times and errors.
- The jacket must protect the disk from external contamination and damage. It should also remove microscopic particles of debris from the disk surface before they can damage the disk.

Why you'll never find the best disk bargain in the bargain basement.

While there's little apparent difference between other disks and Accutrak, the performance differences can be substantial. Simply stated, an Accutrack disk is premium priced. But the protection it gives your information; the reliability it provides to your operations; and its substantially longer life make it the best disk buy. After all, the real cost of your operations is constructing and processing the data stored on the disk - not the disk itself. It doesn't make sense to trust that data to anything but the best disk. Accutrack.

BASUG BULK BUYING — 50% off PRICE FOR 10s

£17:50 incl FREE LIBRARY CASE !

(ADD 50p POSTAGE / PACKING)

BRITISH APPLE SYSTEMS USER GROUP

P.O. Box 174, Watford WD2 6NF.

THE SOFTWARE LIBRARIES

By John Rogers

```
stop press!stop press!stop press!st
!xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxo
sxxxxx SOFTWARE UP FOR GRABS xxxxxp
sxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx!
exxxxx LIBRARIES REORGANISED xxxxxp
rxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxr
p pots!sserp pots!sserp pots!!!!sse
```

OK, now I hope that I have your attention (I must have else you would not be reading this !), please read carefully what follows if you intend to use the software libraries in any way. The revisions within this article take effect as from the 1st September 1981 and supercede the arrangements outlined in previous articles.

All charges in this article are 'all in' rates, i.e. they include postage, packing etc. Tape users will have exactly the same service as disk users except the charges, which will be shown in <> brackets. I will inform tape users of their current balance with each order.

We are sorry that these revisions are being made, but feel that they are necessary to give a more streamlined and efficient service. We also hope that the credit based system will keep up the flow of programs, particularly original material.

The Software Distribution Library (SDL), operates just as if you were buying software from any normal shop- you pay for it. The only differance is that the money you pay is for the disk and postage, not the software content. For this service there will be now one rate only of 3.00 pounds per disk <1.20> (but see the credit calculation formula). This means that the service for copying onto your own disk will, unfortunately, cease with one exception, see the credit calculation formula. The programs in this library (SDL) have come from two sources, 1) given by other user groups, and 2) the more popular programs submitted by members of BASUG .

The Contributed Software Library consists of programs submitted by members of BASUG, that have not progressed into the SDL, see later for programs that have been transferred. The programs in this library (CSL) can only be accessed if you are in credit. The calculation of your credit is shown below.

Credit Calculation Formula

existing balance + value of submitted program(s) = usable balance 1
usable balance 1 - value of programs from SDL = usable balance 2
usable balance 2 - value of programs from CSL = current balance

Notes

1) The calculation must be done in the order shown above.
2) The maximum debit AT ANY POINT shall not be greater than 200 units.
3) The value of the CSL programs will be as detailed in the latest catalog.
4) Refer to table of values for the calculation of submitted programs.
5) ALL SDL programs will be on a BASUG provided disk.

IF IN CREDIT - ie. greater than 50 units at usable balance 1, a choice (or combination of) the following may be requested :-
1) Purchase of SDL disk for £2.00 <£0.80> and 100 units deducted from account.
2) Purchase of SDL disk for full price (£3.00) <£1.20> and 50 units given to account.
3) Maximum of 4 programs from CSL put onto contributors' disk - free, and program values deducted from account.
4) Maximum of 4 programs from CSL put onto disk supplied by BASUG - media value (disk £1.80, tape £0.40) and program values deducted from account.

IF NOT IN CREDIT - only the service of purchasing whole disks from the SDL at the full cost (£3.00) <£1.20> will be available, no deduction will be made from your account.

Exchange rate table

TYPE	VALUE
BASIC	= 1
Machine code	= 3
Text files	= 1
Apple writer file *	= 0.5
Binary graphics file **	= 0.25
Written text * per page	= 5

(NB. all values are ratios to 1 disk sector, apart from the written text)

* eg. program documentation - a page of written text is taken to be 1 page of A4 paper, hand-written or typed text or drawings.

** File(s) created as a demonstration from a program in the submitted suite.

I hope that the operation of the software libraries is understandable, but if you find it neccessary to write in asking for clarification, please make your query as specific as posible. If you feel like a little program writing, then I have a competition for you. There will be a prize of two disks of your choice from the SDL to the person who writes the easiest program to use that calculates CSL credit based on the above formulae and rules.

THE BEST OF C.S.L

No. 20 GAMES

*A 003 HELLO
*A 043 APPLE NUCLEAR PLANT
*B 034 BASUG LOGO
*A 011 GALAXY WAR
*A 020 SPACE LANDING
*A 067 SUPER STAR TREK
*A 022 STAR TREK INSTRUCTION
*A 056 LOST DUTCHMAN'S GOLD
*A 013 COLLISION
*I 018 SPACE INTRUDERS A
*I 018 SPACE INTRUDERS B
*A 017 BRANDS HATCH
*I 007 APPLE ANIMATION
*A 031 MINI GOLF#1

THE BEST OF C.S.L

No. 21 GRAPHICS

*A 024 APPLE SKETCH
*A 007 TEXONGRAF
*B 003 GRTP
*A 015 APPLE/ITT RESULTS PLOTTER
*I 010 ANIMATION
*A 008 CIRCLES & ELIPSES
*A 005 COSINE DECAY CURVE
*A 014 TRIVIA
*A 003 COLOR 21
*T 002 COLORS
*I 003 SCRUNCH
*B 002 SCRUNCH.B
*A 002 GR SCRN CLR]
*I 002 GR SCRN CLR >
*A 003 HI RES COLORS
*A 008 HI-RES SCREEN
*B 009 HI-RES SCREEN.S
*B 034 BASUG LOGO
*A 004 HARMONOGRAPH
*A 002 HELICAL
*I 004 HI-RES DUMP (IP225)
*B 003 HI-RES DUMP.B (IP225)
*I 008 PG 2 GR DEMO
*A 014 LO-RES DEMO
*A 012 MAGIC PAINTBRUSH
*A 005 PROJECTILE PLOT
*A 003 TRUE COLOUR
*B 002 TRUE COLOUR SHAPE TABLE
*A 010 UNION JACK
*A 007 APPLE THREE-D SHAPES
*A 004 HYPERBOLOID
*A 004 SQUARE SPIRALS
*A 020 THE APPLE PAINTBOX
*T 016 PAINTBOX INSTRUCTIONS

THE BEST OF C.S.L

No. 22 UTILITIES

*A 003 HELLO
*B 005 CONCORDANCE NIBBLE
*B 005 CONCORDANCE &-RUN
*A 005 CONCORDANCE &-HIMEM
*B 020 FID
*A 005 PAGESAVER
*B 003 SCREEN
*A 005 DISK MAP
*A 006 PET LOADER RAM
*A 006 PET LOADER ROM
*A 006 PET READER
*A 003 PET SAVER
*A 073 PET NOTES
*T 002 XFR PET PROGRAMS
*B 002 AUTONUMBER.B
*A 002 AUTO NUMBER
*I 014 DISC ACCESS UTIL
*A 005 HIDE
*I 005 MON $ WRITER >
*A 005 MON $ WRITER]
*B 003 SPLIT CAT/SEC FREE
*I 002 SPLIT CATALOG
*I 011 SPLIT CATALOG.ORIG.BASIC
*A 003 TRANSPARENT
*A 012 FP LISTER
*T 002 FP LIST
*A 003 NFC
*B 006 NFC HELLO1.S
*B 002 NFC HELLO1.O
*B 004 NFC HELLO2.S
*B 002 NFC HELLO2.O
*A 009 LINE NUMBER XREF
*B 002 STEP-TRACE
*A 009 AMPERSORT.DEMO
*B 006 AMPERSORT.OBJ
*B 029 PUFFIN
*A 005 PRINT USING
*B 002 PRINT USING.OBJ
*T 007 AMPER-READER.FILE
*B 004 AMPER-READER
*A 018 AMPER-READER.DEMO
*B 034 BASUG LOGO

THE BEST OF C.S.L

No. 23 GENERAL

*A 021 RANDOM DRILL
*A 038 BASIC BEHAVE
*A 036 DATA MANAGEMENT
*A 009 MULTI-BASE CALCULATOR
*A 004 TYPING TUTOR
*A 054 TOUGH ENHANCED (LOWER CASE)
*A 044 TOUGH ORIGINAL (LOWER/CASE)
*A 044 TOUGH ORIGINAL (UPPER CASE)
*A 002 TOUGH INTRO
*A 043 TOUGH FINAL 1.2
*A 008 CALENDAR
*A 058 AIM(ENHANCED VERSION 2)
*B 034 BASUG LOGO
B 002 SEARCH

DISK No. 24

DOS SYSTEM MASTER

DOS 3.2.1 PLUS

*A 002 HELLO
*A 015 COPY
*A 028 FINANCE I
*B 009 UPDATE 3.2.1
*A 026 PENNY ARCADE
*A 047 LEMONADE
*B 003 CHAIN
*A 009 COLOR DEMOSOFT
*A 028 LITTLE BRICK OUT
*A 003 MAKE TEXT
*A 003 RETRIEVE TEXT
*A 010 EXEC DEMO
*A 010 RANDOM
*T 003 APPLE PROMS
*A 039 RENUMBER INSTRUCTIONS
*A 014 RENUMBER
*A 006 BRIAN'S THEME
*A 058 HOPALONG CASSIDY
*A 051 PHONE LIST

DISK No. 25

DOS SYSTEM MASTER

DOS 3.2.1 STANDARD

*I 002 HELLO
*I 053 APPLE-TREK
*I 018 ANIMALS
*B 009 UPDATE 3.2.1
*I 014 COPY
*I 009 COLOR DEMO
*I 053 BRICK OUT
*I 026 SPACE WAR
*I 050 THE INFINITE NO. OF MONKEYS
*I 051 COLOR SKETCH
*I 053 SUPERMATH
*I 026 APPLEVISION
*I 017 BIORHYTHM
*I 027 PINBALL

Lastly, and in no way least, will you please, please, please write ALL correspondence regarding the software libraries ON A SEPARATE SHEET OF PAPER from any other correspondence and orders, it makes the administration ten times easier, faster and less prone to errors.

As a postscript, I have in my possession a number of disks that have had programs contributed to the CSL on them. If you are the proud owner of such a disk, then please drop me a line telling me what to do with it.

This article includes details of ten disks added to the SDL, including four consisting of programs transferred from the CSL.

DISK No. 26

APPLE CONTRIBUTED SOFTWARE VOL 1

*I 007 HELLO
*I 005 CHR$ FUNCTION
*I 010 COLOR MATH
*I 029 PINBALL
*I 009 OTHELLO
*I 043 APPLESOFT
*A 007 HEX CONVERTER
*I 005 CATCH
*B 007 COPY.OBJ
*I 007 COPY
*I 008 TWENTY-THREE BRICKS
*I 024 SEVEN
*I 005 CURVES
*I 018 TOWERS OF HANOI
*I 022 NIGHTMARE $6
*I 012 SINK THE SHIP
*I 045 YAHTZEE
*I 009 MASTERMIND
*I 006 MORSE CODE
*I 018 SLOT MACHINE
*A 022 BONE TUMOR DIAGNOSIS
*I 019 BLACKJACK
*I 019 HAMMURABI
T 001 BASENAMEFILE

DISK No. 27

APPLE CONTRIBUTED SOFTWARE VOL 3

*I 007 HELLO
*I 007 COPY
*B 007 COPY.OBJ
*I 043 APPLESOFT
*I 022 INTERCEPT
*A 013 AIRFOIL
*A 029 MICROLISP
*I 010 SHOOTOUT
*A 010 HI-RES CHARACTER DEMO
*B 003 HI-RES CHARACTER GENERATOR
*B 006 CHARACTER TABLE
*I 026 APPLE VISION
*I 014 ENGINE
*A 038 FILE CABINET
*B 006 INTEGER HI-RES
*I 037 KALEIDOSCOPE

DISK No. 28

WASHINGTON APPLE PIE VOL 7

*A 002 WASHINGTON APPLE PIE VOL 7
*I 012 COLOR MATH
*I 019 CONVENTIONS
*A 008 ECHOCARDIOGRAPH
*I 026 FLASH CARD
*I 044 FLASH CODE
*I 006 INTGER INSTRUCTION SET
*I 021 MATH TUTOR
*A 015 MORSE CW
*I 016 MORSE TRAINER
*A 021 NAME STATES
*A 021 NORTHERN CONSTELLATIONS
*I 014 QUIZBUILD
*T 003 QUIZFILE
*I 030 SIMULATION-6502
*A 013 STATES/CAPITALS
*A 019 TITRATION
*I 026 TOP DOWN PROGRAMMING
*I 013 TYPING PRACTICE
*A 002 HELLO
*B 008 MENU

DISK No. 29

WASHINGTON APPLE PIE CONTRIBUTED

*I 035 A TRILLION STORIES
*I 004 ANDROMEDA STRAIN
*I 012 AWARI
*I 007 BAGELS
*I 010 BANANAS
*I 018 BIORHYTHM
*I 010 BLACKJACK
*I 006 COLOR LIFE
*I 019 COLOR TEXT
*I 002 COLOR WORM
*I 005 CRAPS
*I 006 DIGITAL CLOCK
*I 003 DRIP
*I 003 GARYS QUICKY
*I 008 INTRO
*I 004 KALEIDOSCOPE
*I 005 KENO
*I 016 MAD-LIB
*I 008 MASTERMIND
*I 009 MERRY CHRISTMAS
*I 017 MIDWAY
*I 009 MOUSE MAZE
*I 007 NIGHTMARE $6
*I 010 OTHELLO
*I 025 POET
*I 034 POKER
*I 010 ROULETTE
*I 002 SANDYS FOLLY
*I 008 SAUCER WAR
*I 011 SEA CHASE
*I 021 SEVENS
*I 011 SHOOTING STARS $2
*I 016 SLOT MACHINE $2
*I 002 SQUARES
*I 005 TENNIS

---

software

PROFILE - FRANCES TEO

By Tony Williams

New members of BASUG who will shortly receive copies of the "Best of Hard Core 1 and 2" reprint will be doing so thanks to the efforts of a small band of BASUG members, including Frances Teo. Before agreeing to take on this assignment Frances, one of the founder members of the group back in the dim and distant days of November 1980, had some spare time on which to earn her keep and hone up her programming skills. She has now kindly volunteered under duress to step in and provide the data about her work and life for the first in what is planned to be a short but hard-eyed series on "What Makes an Apple User?"

Frances does not come from a family tradition of computing (who does, apart from the whizz kids of Silicon Valley?) but the next best thing - her father was a mathematical engineer working out at Woomera amongst other places. She was educated at Sydney Technical College and did a couple of years of an accountancy course. She has a high school leaving certificate in Maths I, Physics, Chemistry and English. After college she moved to England and served a term of marriage and motherhood - is in fact still married and still a mother. Her professional career began while helping her dentist husband with his practice's accounts, but soon she moved into hotel accountancy - which was where she first came into contact with computing.

The hotel in question had been run by an electrical engineer who was interested in doing something with computers to tackle their accounts which had outgrown the manual methods used until then (1979). Elegantly, he asked Frances to set the wheels rolling, gave her no time off to learn programming but did allow her sufficient office time.

Frances handled this assignment in time-honoured fashion - by reading an ad in the Sunday Times for a Tandy, ringing up for a demonstration. She was given a prompt and superb presentation which convinced her completely. The trouble only came later. The software package with the Tandy was a Tridata Nominal Ledger 300 which worked fine, except that to handle the hotel accounts properly Tridata said that a third disk drive would be necessary - and this would have strapped the hotel's finances too hard. The crunch came, however, with the crunch - when the TRS80 drive started chewing up centre rings of disks and masses of vital data was irretrievably lost.

All this time Frances was second-stringing by computerizing the dental practice's accounts. Ever an avid ad-reader she responded to an advertisement in the British Dental Journal for a one-week Basic Training Assault Course at an hotel in Looe, Cornwall. This was run by a retired couple, he an ex-systems analyst turned chef. (Can there be an affinity between cooking the books and just plain cooking?). The course was not orientated toward either dentistry or cooking, however. While attending this course Frances asked around about other systems to replace her disk-gnawing Tandy and was recommended the Apple, purely by virtue of the reliability of its disk drives and its graphics capabilities.

Since that time the trusty TRS 80 on which she began her programming career has been gathering dust in her back bedroom. She bought the Apple (DOS 3.3, 48k two disk drives) primarily for the dental practice and has since severed her connections with the hotel business. Her present ambition is to develop customized software for dental practices.

Working together with a systems analyst (part-time) Frances has already done the spadework on a dental package to go on the market toward the end of the year. Since she is on the inside looking out, she has been in a good position to decide how to make her product suit real needs instead of pure conjecture. The computer buff might imagine that what is wanted is a some kind of terminal to stand in the surgery on which the assistant enters details of cavities, etc. Not necessary, says Fran. Cards do very nicely, thank you. What the dentist really needs is a comprehensive package covering payroll, expenses etc. in the usual way, but above all, keeping track of dealings with the NHS, approvals for bridges and crowns, etc. In writing software to meet this genuine need Francis also ensures that her system can be operated with the bare minimum of expertise by the typical sixteen-year-old receptionist-assistant who is only likely to stick around for three months.

Francis Teo is trying to grasp the lodestone sought by all programmers that will enable her to turn pure ideas and thought into gold without the need for outside capital investment. With a maximum of some 10000 dental practices in the NHS you don't need Visicalc to weigh up the scope for success. Software sales alone are unlikely to do it and she wisely plans to sell entire systems. If Graham Rubens can exist by catering for football clubs, Frances Teo would seem to have a fighting chance of surviving in the world of gums and premolars. Any thoughts on this from our various dental members? The trick is to stay sane and survive the first year or so until some reward starts coming in.

Pippin's Page ~~~~

Edited for younger readers by Vernon Quaintance

Holiday time is here again and for some of us it gives a chance to spend more time with our Apples than we normally can after doing homework.

Do you get bored during the holidays? – Nothing to do except go boss-eyed playing Space Invaders? Why not try writing something of your own? It is not as hard as it sounds and you can then send your work to Pippin's Page to share it with others.

I have received two disks from Ian Springle, aged 11, of Congleton in Cheshire. They include an editor, a shape designer, and several games. Thank you Ian, I particularly like your high-res graphics logo pages. I am trying the games out at ComputerTown. So far, Echo and Suicide Bombers are proving popular.

Talking of ComputerTown, have you one near you? If so, and you don't already attend, why not visit ComputerTown sometime. There are usually several different machine types for you to try. Remember though that Applesoft has many features not found in other Basics.

In the next Pippin's Page I hope to start describing each of the commands of Applesoft Basic, but to start things off, let's take a guided tour of the inside of our Apple. First switch the Apple off, and then lift the lid.

Along the left hand side is a large metal box. This contains the power supply which provides the right electrical voltages to make all the parts of the Apple work properly.

Across the back are eight long slots. Extra cards plug into these for driving things like printers and disks. In the centre, just in front of these slots, is a large, flat, black plastic box with 40 legs. This chip is the 6502 microprocessor which does all the real work inside the Apple. (The same chip is used in the PET, Acorn and many other computers).

In front of the microprocessor are six medium sized chips. These are the ROM (Read Only Memory). The one on the left is the Monitor ROM. This controls frequently used things like reading the keyboard, and writing to the screen.

The five chips to the right of the Monitor are the Basic Interpreter. This is like a dictionary. It looks up the Basic commands which we type and translates them into the binary (long strings of ones and noughts) wanted by the processor. The other way around, it takes the binary from the processor and translates it into Basic (ie something near to normal English) for us to understand.

In the middle of the computer there is a white-outlined box. This contains one, two or three rows of small chips. These are the RAM (Random Access Memory) which form the main memory of the machine. Each row contains just over 16,000 memory locations (16K). Three rows make the full 48K capacity of the Apple. (RAM is really badly named because access is not at random, but controlled. It is really read/write memory.

The remaining chips do all sorts of odd jobs including providing an internal clock, reading the game paddles, and acting as a sort of telephone exchange to send signals to the right part of the computer.

Right, now we have looked inside the Apple and know where things are, and what they look like, we can put the lid back, switch on and start using the Apple again.

I look forward to seeing some of you at the forthcoming P.C.W. Show. Watch out for a course in Basic starting next time.

Happy Apple Computing!

Expediter
"The" Applesoft Compiler ?

By Mike Walker

The On-line Systems Compiler converts a "standard" Applesoft program into a single line machine code call.

The compiled program usually requires several times the memory used by the Basic program and typically runs at three times the speed, depending very much on the type of program.

To compile a program all that is required is to load the Basic program and Brun Expediter ..... After 10 seconds you are asked whether you wish to accept the "Default Parameters" or set up your own.
You can change where the program will reside, the upper limit for Local and Global variables, string length and reserve memory. The reserve memory option enables you to prevent the compiled code occupying memory needed for the program itself, for instance page one of hi res graphics.

The Default Parameters are.....

Program Counter - $1100
Local Variables - $9600
Global Variables - $9600
String Length - 40
Line Trace - on
Reserve Memory - no
Display Adresses - yes

The Display Adresses option gives a line by line analysis of the generated object code as it compiles e.g.

LINE:1 PC:$1120 VAR:$9600 SRC:$11B2

SRC tells you how much source (basic prog) remains to be compiled.
There is a "Runtime" package containing general purpose subroutines used by Expediter which occupies $800 - $10FF. This package is automatically saved with the compiled code

An interesting feature of Expediter is the "line trace" which allows the compiled program to give normal Basic line error messages from the compiled code thus aiding debugging as you can go back to the Basic program line, fix the problem and recompile. Line trace is one of the options available at compile time and should be used initially to check out the code but not for the final code as it takes 7 bytes per Basic line therebye slowing down the code.

Unfortunately version 2.1 which I was using has a few minor bugs in it, the current version 2.2 should have corrected these. The "Chicken animation" from the "BASUG" disk suffered from one of these bugs... when the compiled version was run the chicken moved at almost three times the speed which was very impressive considering it had lost a leg in the compilation !!
This bug is caused by the applesoft DRAW 1 command which causes a "SYNTAX ERROR"during compilation, DRAW 2, 3 etc. are not a problem.

Another interesting "problem" is that the BASUG disk HELLO runs at half speed !! This appears to be caused by line 500 which prints strings onto the screen backwards using LEN$ & MID$ functions. ON-LINE SYSTEMS are looking into it.

A standard Basic Shell Sort runs five times faster, a Binary Sort at four times faster.

Expediter makes extensive use of pre-existing routines within the Applesoft ROM,so that many functions of Applesoft will run at normal speed, Applesoft mathematical functions for instance, are not any faster.

One important restriction is the inability to DIMension an array using DIM(N)
The N must be an integer not a variable.

CONCLUSIONS

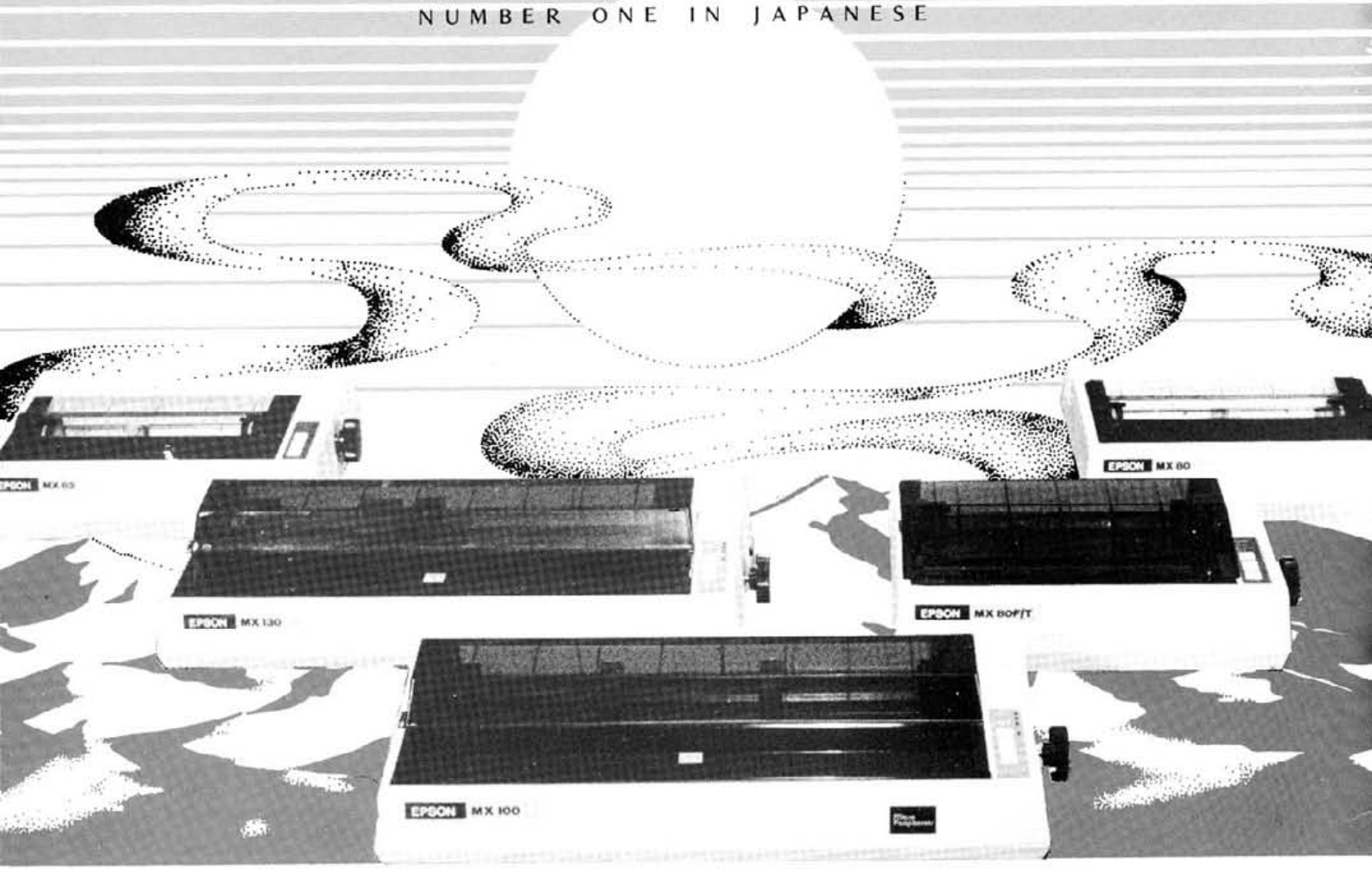
A nice program , easy to use, very flexible, gives 3-5 times speed, sometimes much more, not ideal for speeding up existing material as it may require debugging although it worked in 60-70% of the programs I tried.

It is particularly useful for increasing Sort speeds and Graphics where it can make a substantial improvement to the operation of the program.

My thanks to S.B.D. SOFTWARE. for the loan of EXPEDITER.

ICHIBAN

NUMBER ONE IN JAPANESE



Highest possible quality + lowest possible cost = EPSON +MICRO PERIPHERALS
for the best of both worlds.

	Matrix			Paper Handling		Speed		Max. Paper Width		High Res. Graphics & Italics
	7 x 5	9 x 7	9 x 9	Friction	Tractor	80 CPS	135 CPS	10"	15½"	
MX80 T			●		●	●		●		
MX80 FT/1			●	●	●	●		●		
MX80 FT/New Type 2			●	●	●	●		●		●
MX82			●		●	●		●		●
MX100			●	●	●	●			●	●
MX130		●		●	●		●		●	●

The above machines have many more features including interfaces for Apple, PET (with PET Graphics), TRS80, Sharp, NEC, Hitachi, Nacom, Acorn, Super Brain, Video Genie, BBC Micro etc, some have correspondence quality printing and multiple character sets including international languages. Ring Ian today for full details and specifications and printout samples. All machines usually ex-stock with next day delivery plus 12 months no-quibble guarantee.

Wholesalers and Distributors of quality Japanese Micro Products.

Micro Peripherals ltd.

61 NEW MARKET SQUARE, BASINGSTOKE, HANTS. Telephone: 0256 56468

Europe's leading Epson Distributor and Supplier. Trade Sales - 0256 56469

JAPANESE OFFICE: 101 Abe Bldg. 4F, 2-42 Kanda Jinbocho, Chiyado-ku, Tokyo, Japan.